Introduction to Geoprocessing in ArcGIS 9

ND GIS Users Conference – Bismarck, ND
October 24, 2006

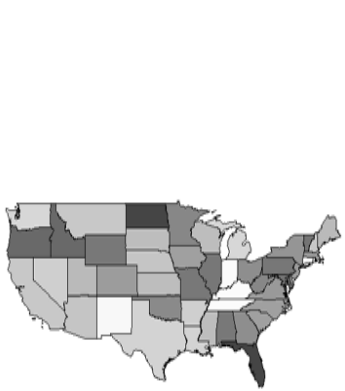
Travis Saladino
ESRI-Minneapolis

Workshop objectives

- **Introduce Geoprocessing in ArcGIS:**
  - **Tools (dialogs)**
  - **ModelBuilder**
  - **Scripting**
  - **Command Line**
  - **Advanced topics**
- **Ask questions and participate in discussions**

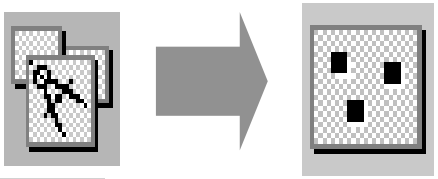
What is geoprocessing?

- **Geoprocessing is the processing of geographic information, one of the basic functions of a GIS**
- **Perform a variety of geographic based tasks**



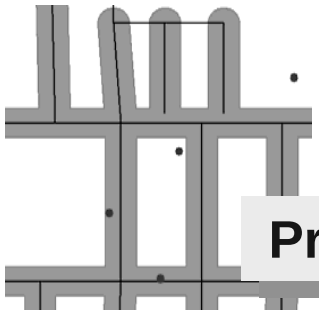
Projections

CAD **GDB**



Conversion

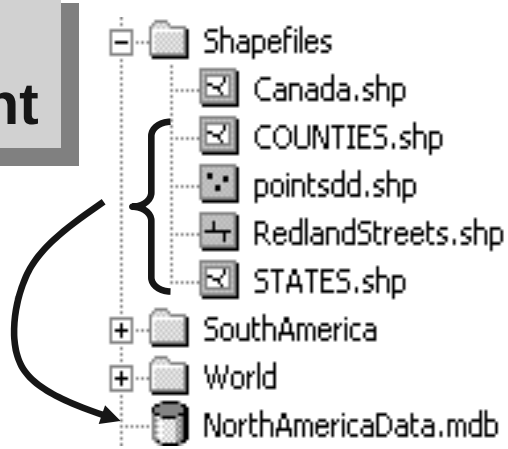
Database



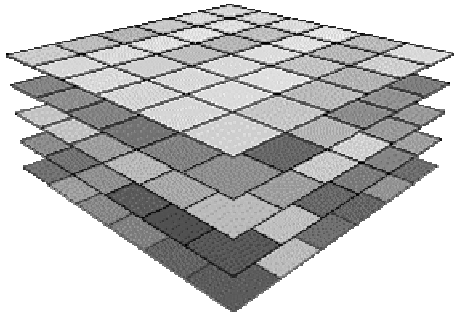
Spatial Analysis

Proximity?

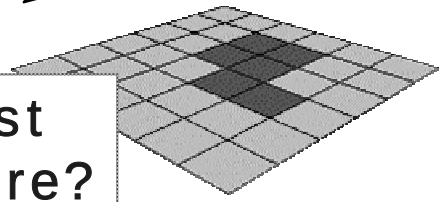
Data Management



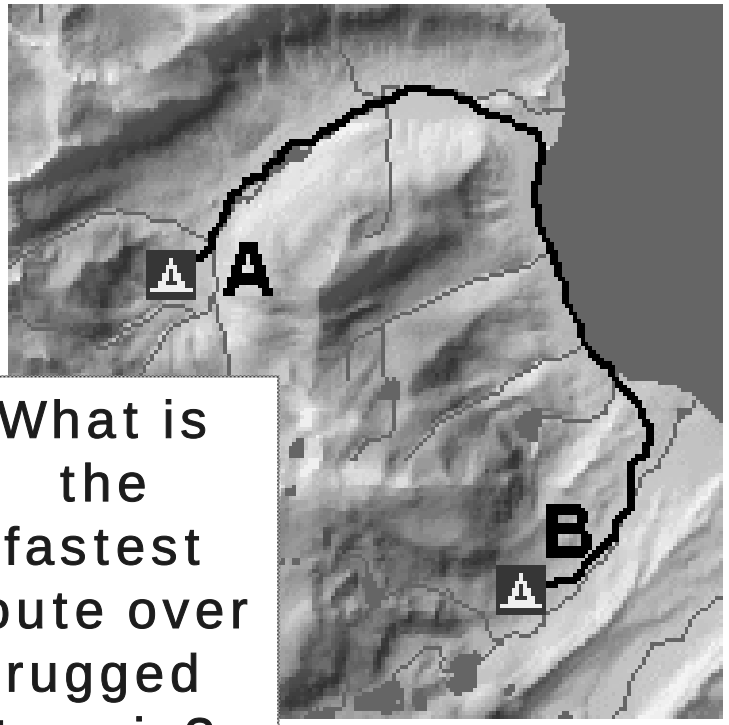
Geoprocessing answers spatial questions



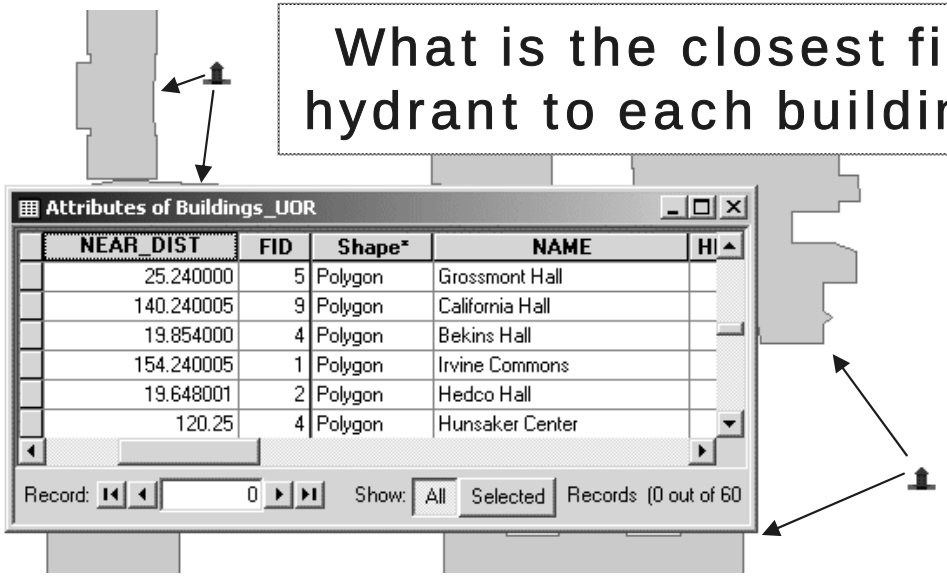
Census Block Groups
Distance to Roads
Land Cost
Distance to competitors



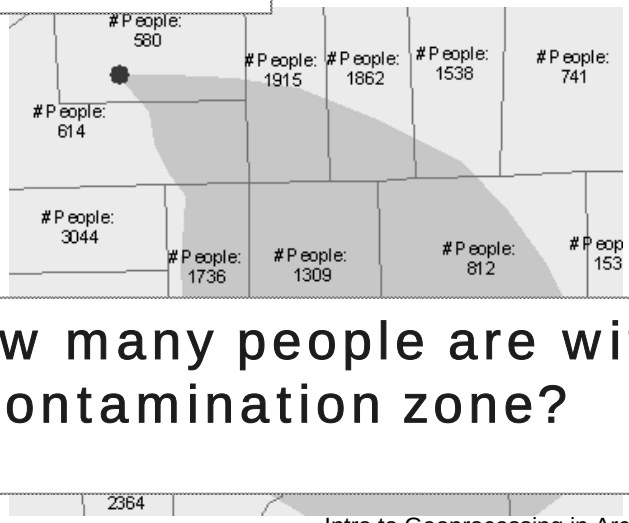
Where is the best location for a store?



What is the fastest route over rugged terrain?



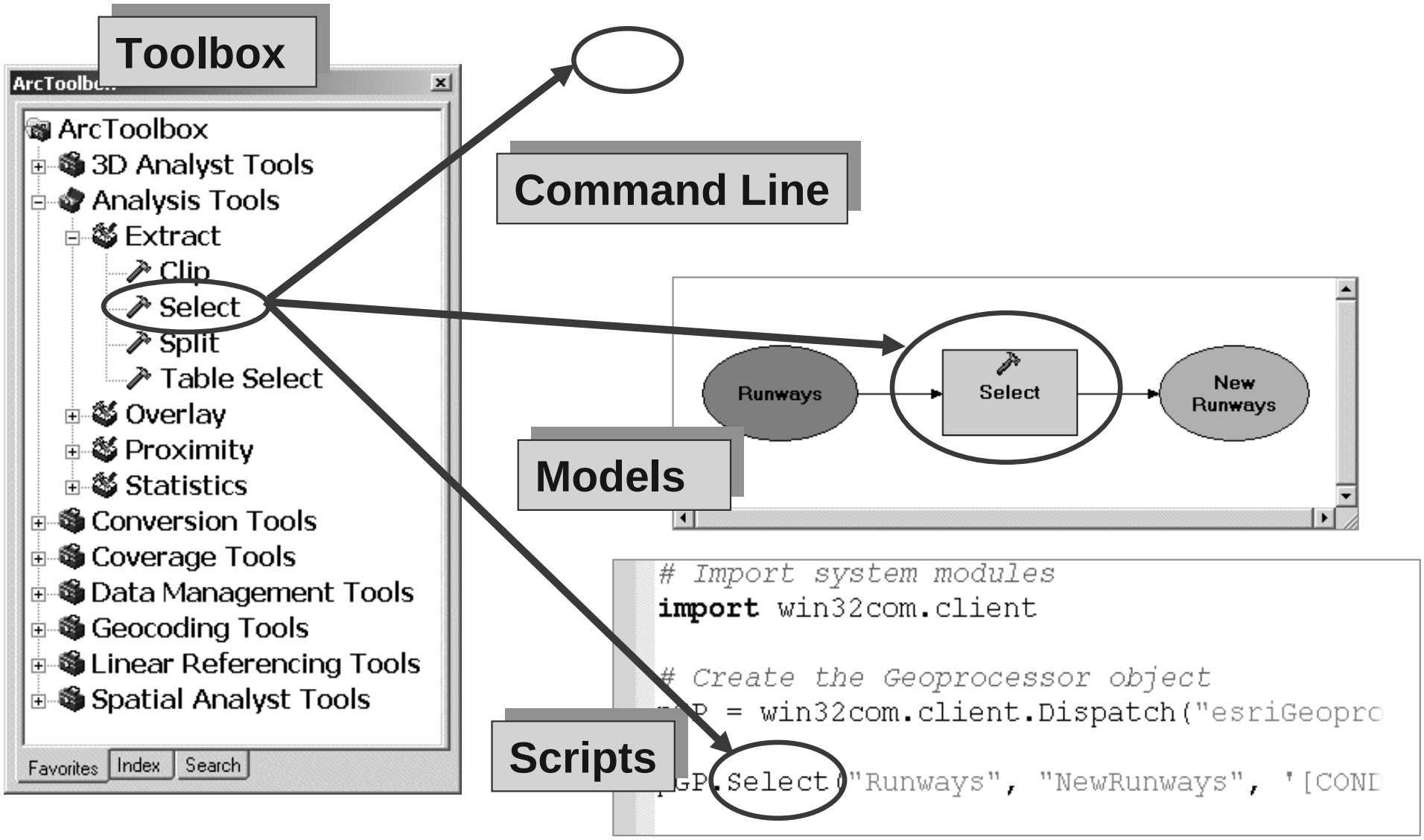
What is the closest fire hydrant to each building?



How many people are within a contamination zone?

Geoprocessing at ArcGIS 9

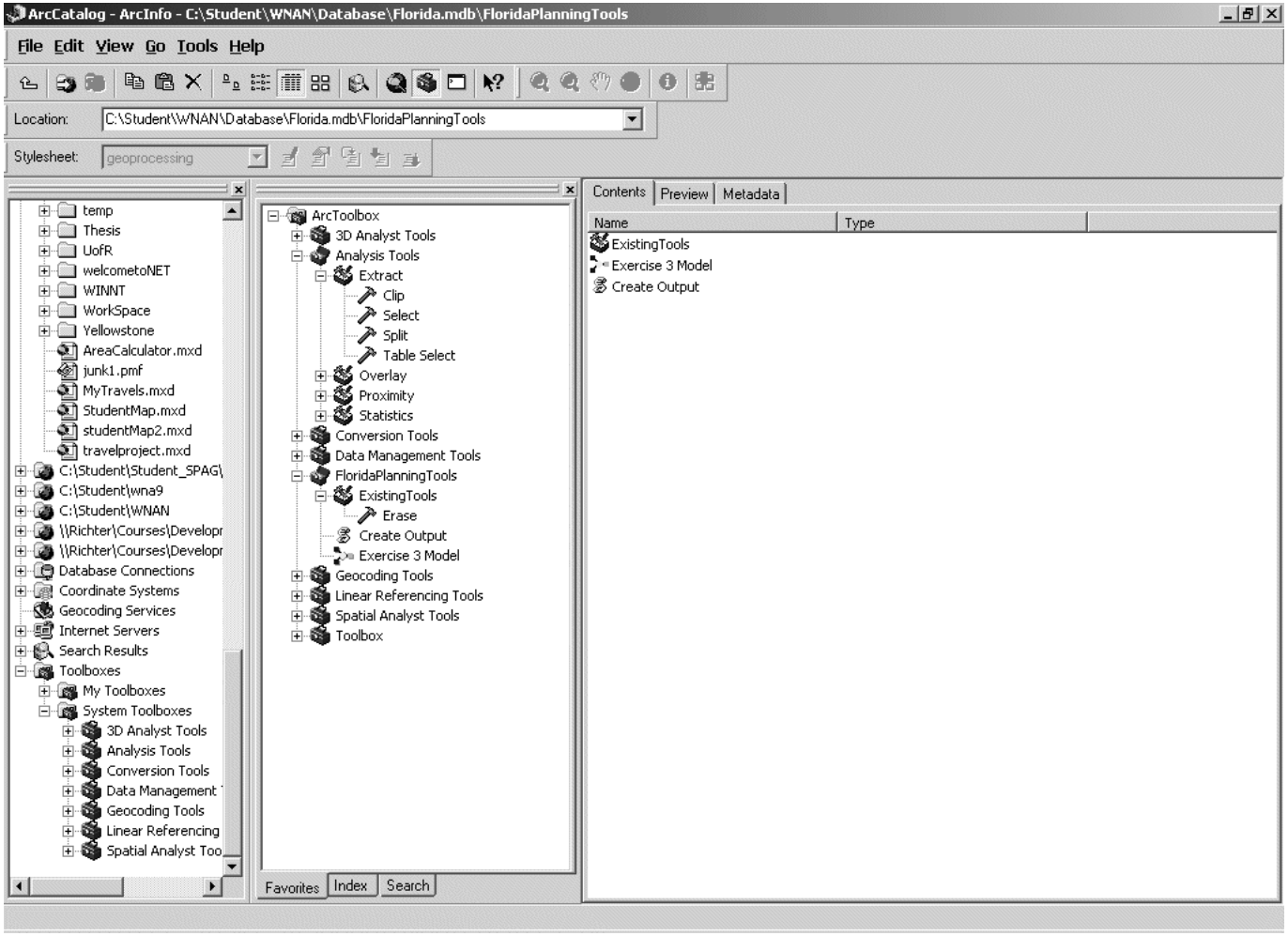
- **Geoprocessing framework available with ArcGIS 9**



ArcToolbox

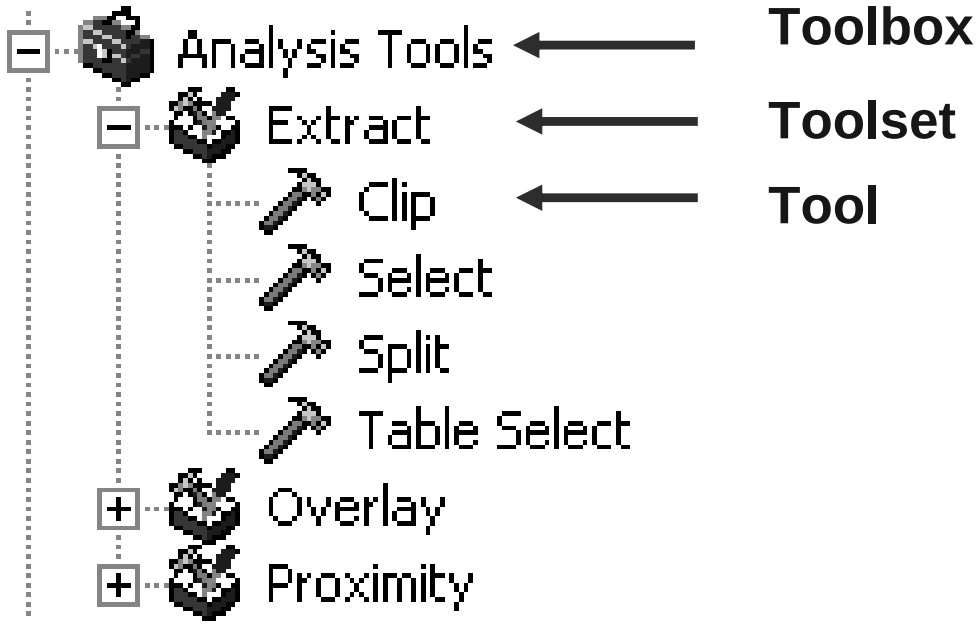
Accessing the ArcToolbox window

- A dockable window with toolboxes
  - Obtained in ArcMap, ArcCatalog, ArcGlobe, or ArcScene
  - Tree view in ArcCatalog
  - No longer a separate application



Toolsets and tools

- **Toolbox: Container for tools and toolsets**
- **Toolset: Logical container of tools and other toolsets**
- **Tool: Single geoprocessing operation**
  - **System tool, model, script, custom built tool**
  - **All tools look and act the same**



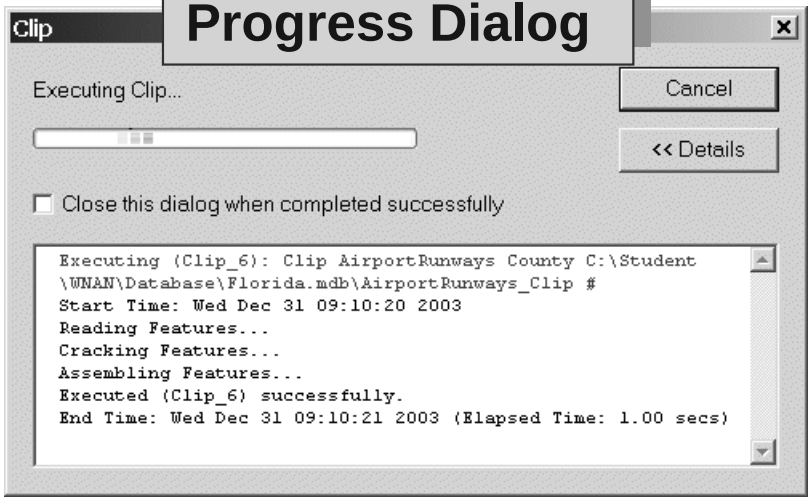
Executing tools

- Double-click any tool to activate the tool dialog
- Specify parameters
- Click OK to run
- Messages appear in progress dialogs and Command Line window

Tool Dialog

Command Line

Progress Dialog



Tools and licensing

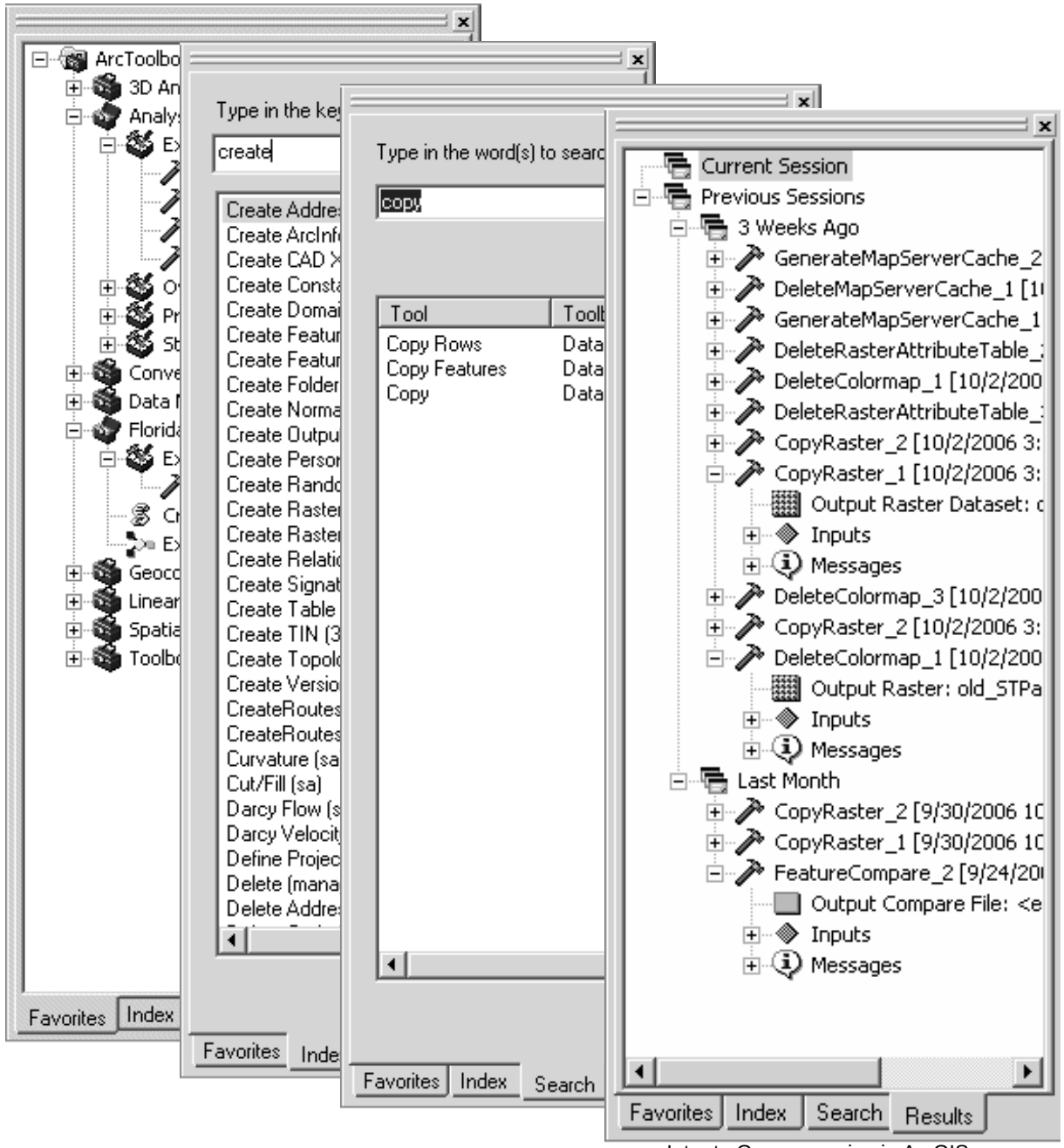
- **Ability to use ArcToolbox, command line, models, scripts is included at all license levels**
- **The amount of tools vary depending on license level and extensions you have available**

- **License Levels**
  - **ArcView = 102 tools**
  - **ArcEditor = 104 tools**
  - **ArcInfo = 251 tools**
  - **9.2 has 81 new core tools**
- **Extensions**
  - **Spatial Analyst extension = 158 tools**
  - **3D Analyst extension = 45 tools**
  - **Network Analyst extension = 16 tools**
  - **Geostatistical Analyst = 1 tool**
  - **9.2 Adds 40 new tools to extensions**

** ***At 9.2 a total of 582 tools with ArcInfo and all extensions***

Toolbox tabs

- **Favorites**
  - **Most frequently used tools**
- **Index**
  - **Alphabetical listing of all tools**
- **Search**
  - **Search tools by name or keyword**
- **Results – New at 9.2**
  - **Historic view of GP results**



Creating your own Toolbox

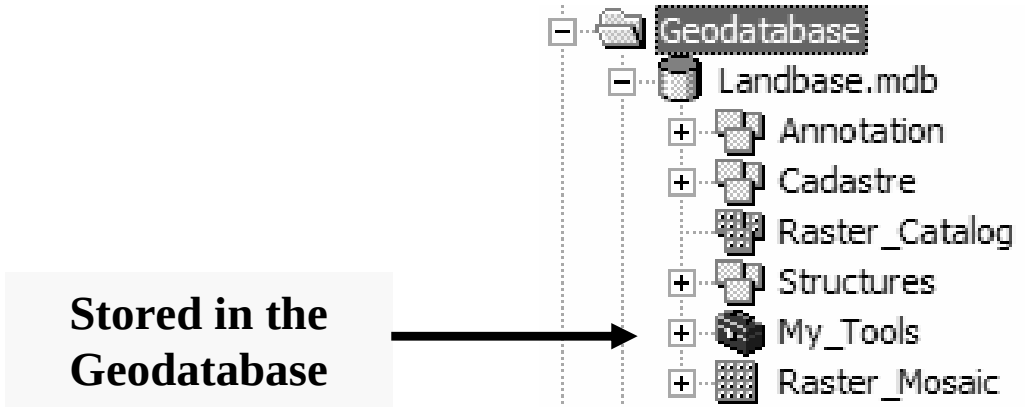
- Create new toolboxes in ArcToolbox or ArcCatalog
  - Used to store models, scripts, custom tools, or a shortcut to frequently used tools

- Where can Custom Toolboxes be saved?

  1) My Toolboxes
     - Created in ArcToolbox or Catalog Tree
     - Stored as .tbx in your user profile directory

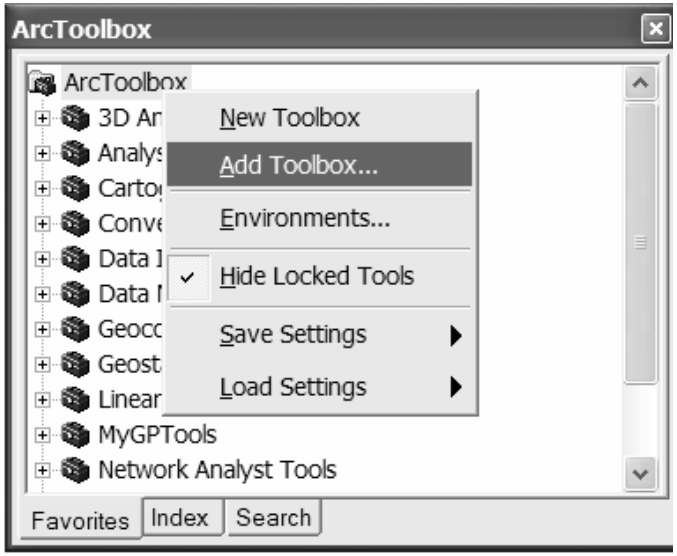
  2) Folder
     - Created in Catalog Tree
     - Stored as .tbx in folder

  3) In a Geodatabase
     - Created in Catalog Tree
     - Stored as table in GDB

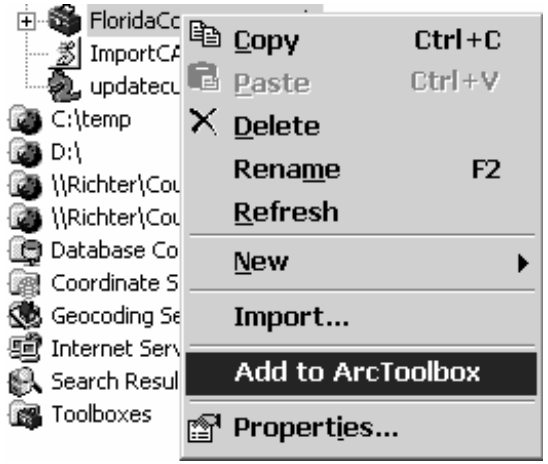


Adding a toolbox to ArcToolbox

- In ArcToolbox
  - Browse for toolboxes
  - Add Toolbox dialog
  - Navigate to toolbox and Add



- In ArcCatalog
  - Add any toolbox by right-clicking
  - Copy and paste into ArcToolbox



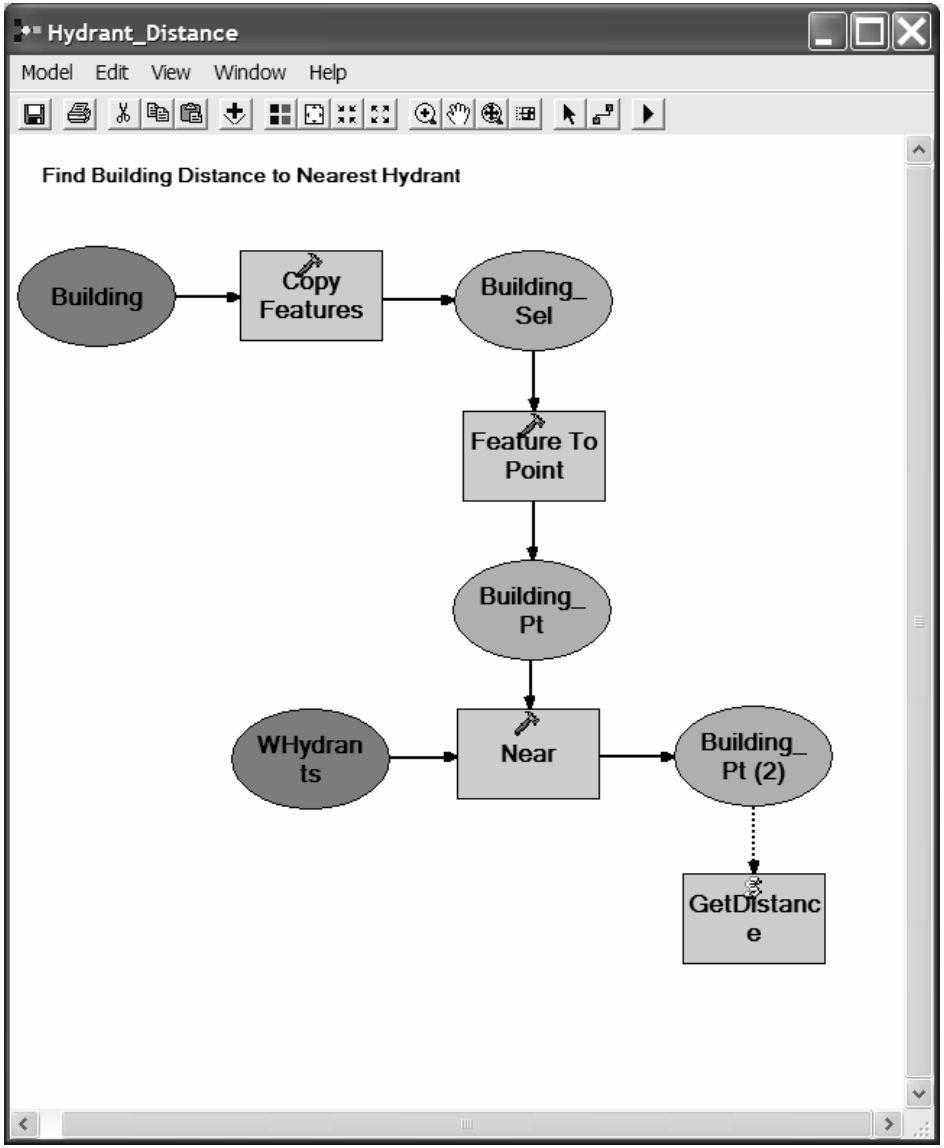
Creating your own tools

- Three ways to create a tool
  - ModelBuilder
  - Scripts
  - Custom-built tool with ArcObjects
- All tools act and behave the same
  - Execute as a dialog
  - Accessing documentation

ArcGIS ModelBuilder

What is ModelBuilder?

- **ModelBuilder is a window tool with a graphic environment where geoprocessing models can be built**
- **ModelBuilder is included with all ArcGIS Desktop license levels**
  - ***(ArcView, ArcEditor, ArcInfo)***
- **The ModelBuilder window consists of:**
  - **A display window in which you build a diagram of your model**
  - **A main menu and a toolbar that you can use to interact with elements in your model diagram**
  - **You can run the model from within the ModelBuilder window or from its dialog box.**



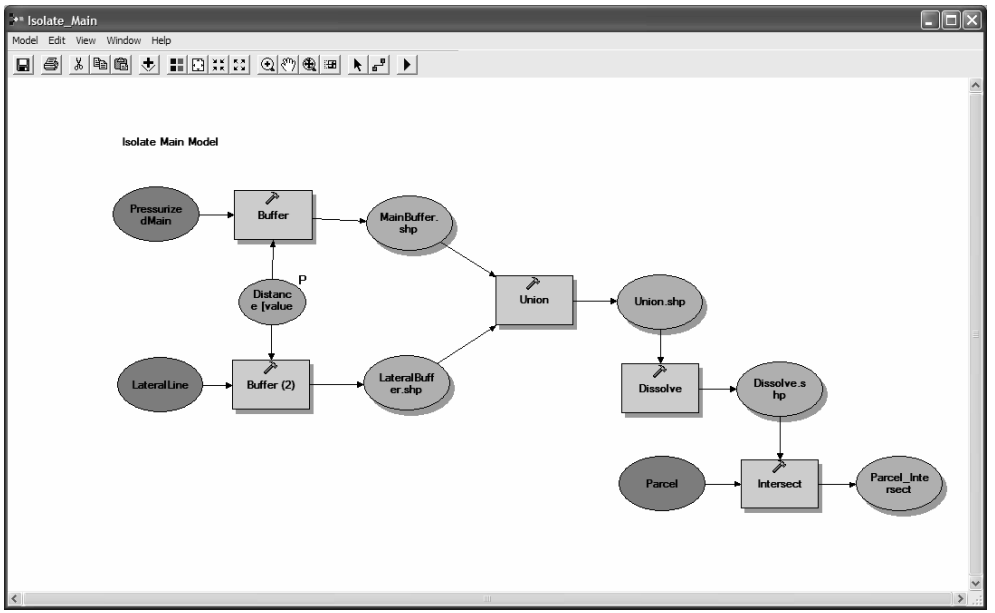
ModelBuilder

Types of Models

- Repetitive Tasks → Execute a series of frequently used tools
- Suitability Models → Use to find a best location
- Process Models → Show the landscape as conditions change

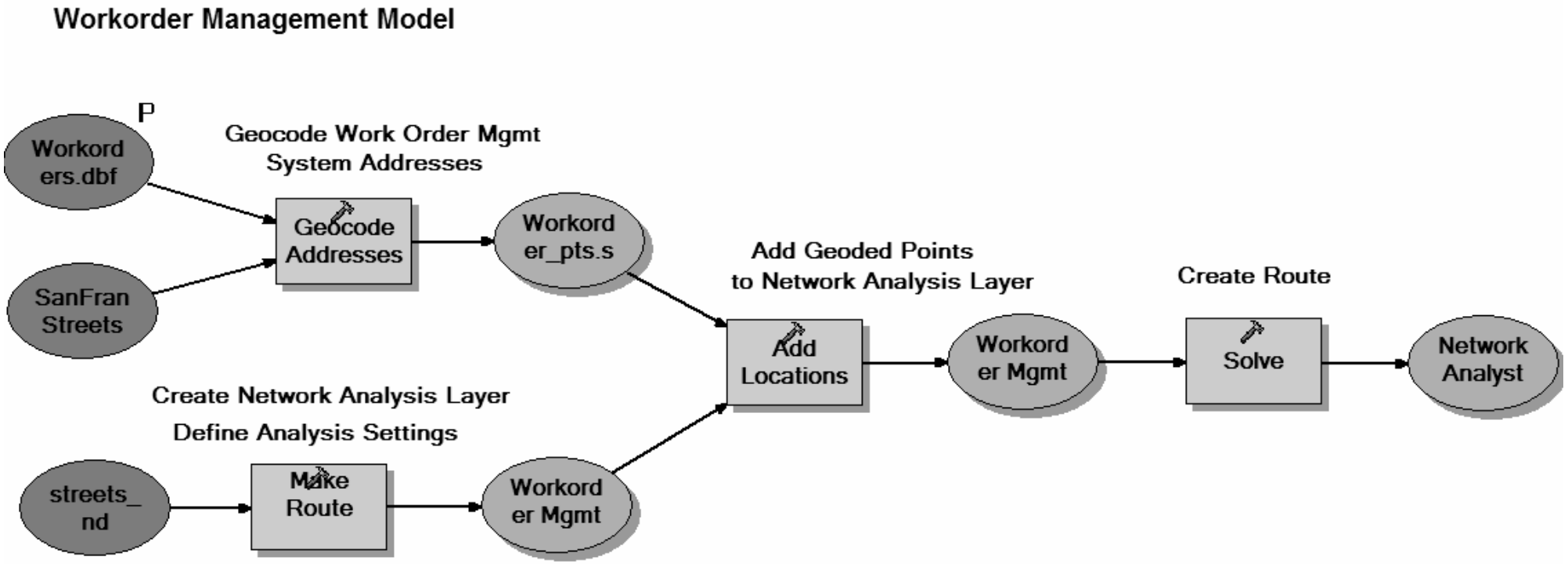
Why use ModelBuilder?

- Faster Analysis
- Ability to create complex models
- Ability re-execute the same model and change parameters
- Graphic documentation of work (metadata)



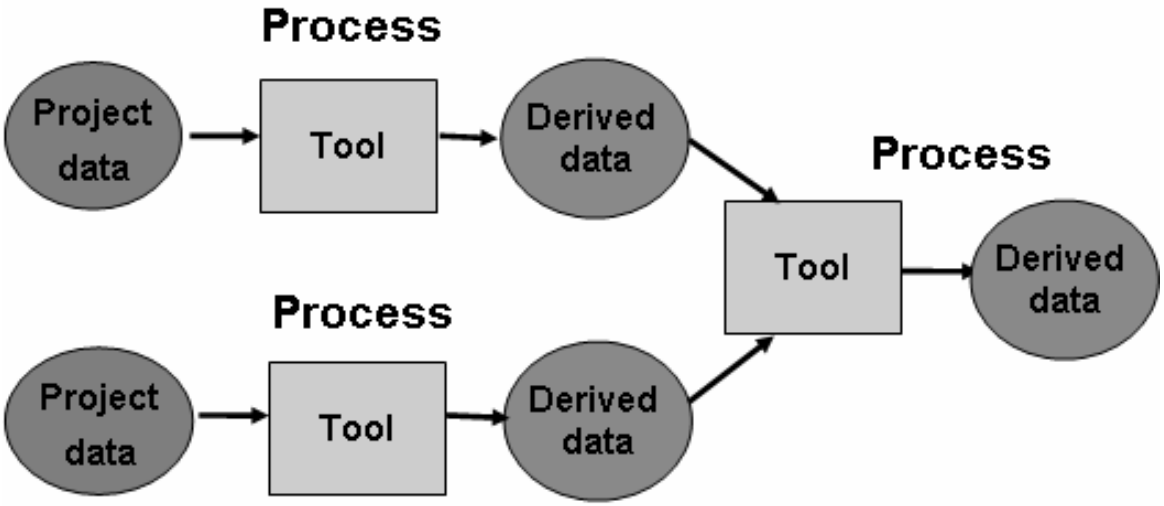
Example Model

- **Find the optimal driving route for a table of workorders exported out of workorder management system**



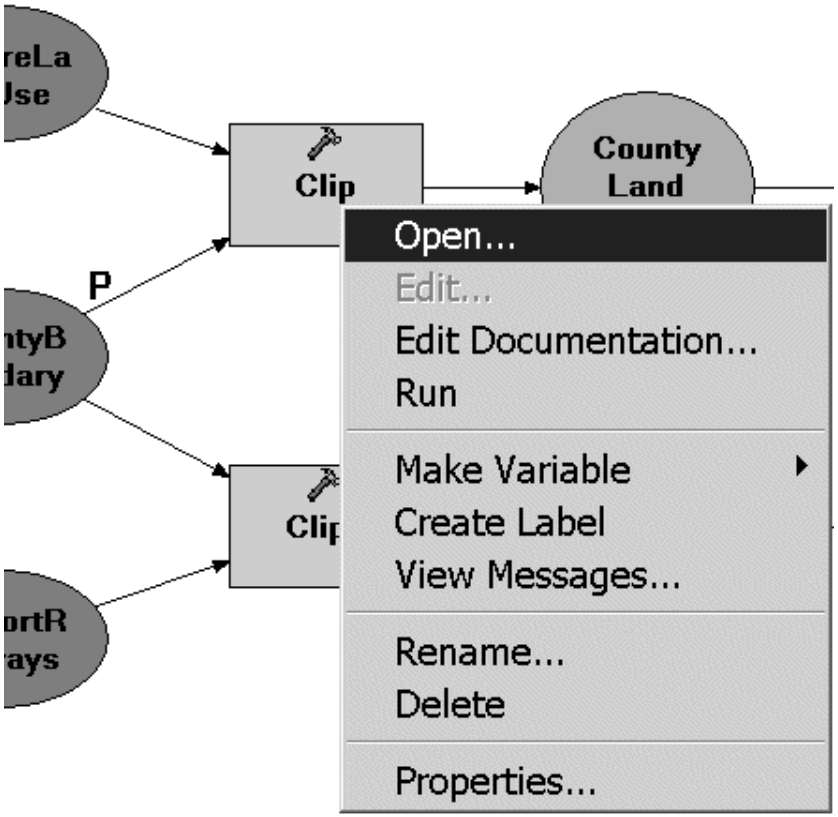
Model elements

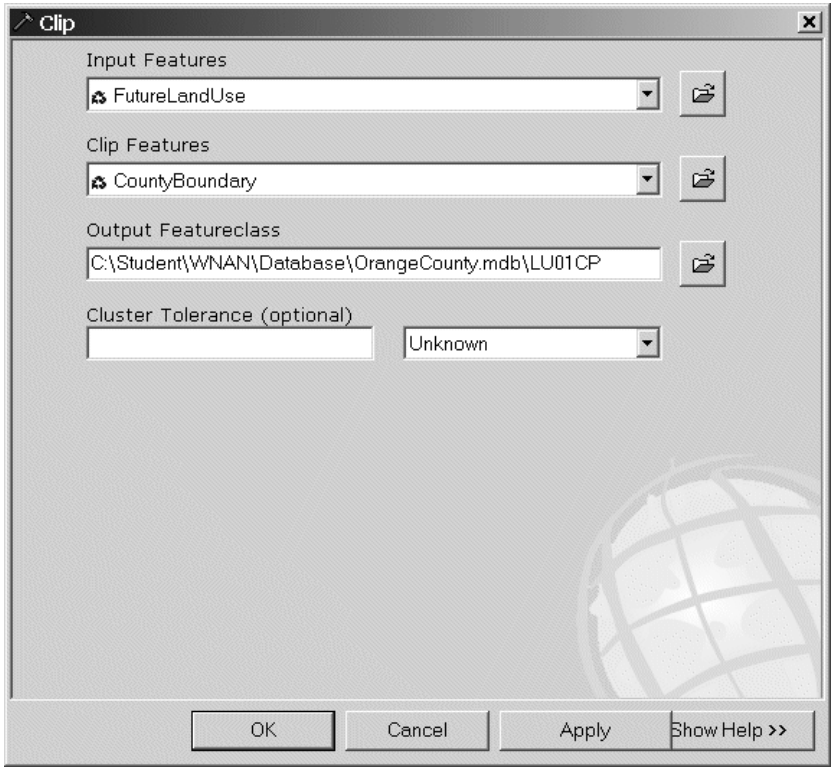
- **Project data: Data that exists before model is run**
  - **Blue oval**
- **Tool: Operation performed on input data**
  - **Yellow-orange rectangle**
- **Derived Data: Output data created by a function**
  - **Green oval**
- **Process: Set of elements**
  - **Run one process at a time or all at once**



Tools within a model

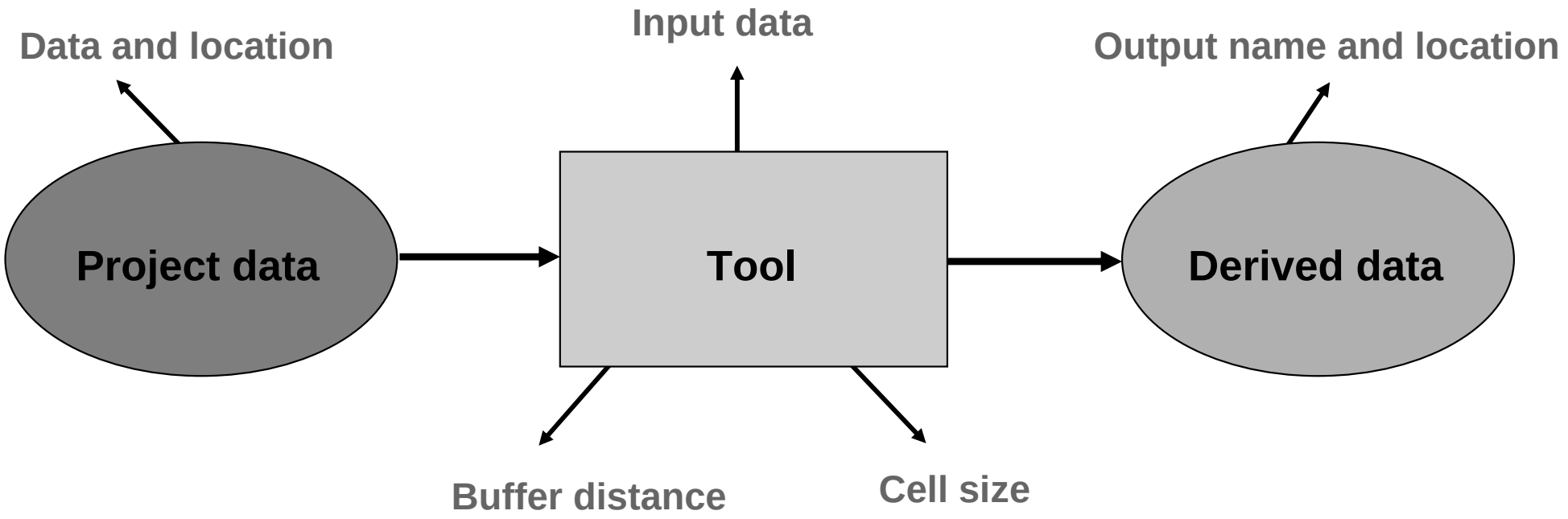
- **Right-click or double-click to obtain parameters**
  - **Same dialog as tools from a toolbox**





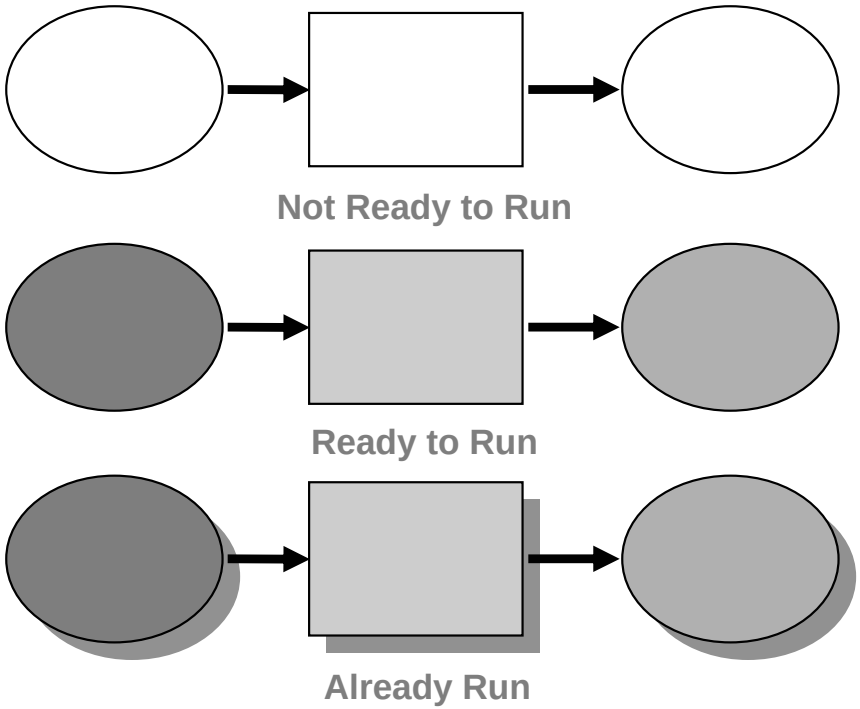
Parameters

- **Input/Output data and values for a tool**
- **Used for running model as dialog**
- **Right-click model element and choose to create parameter**



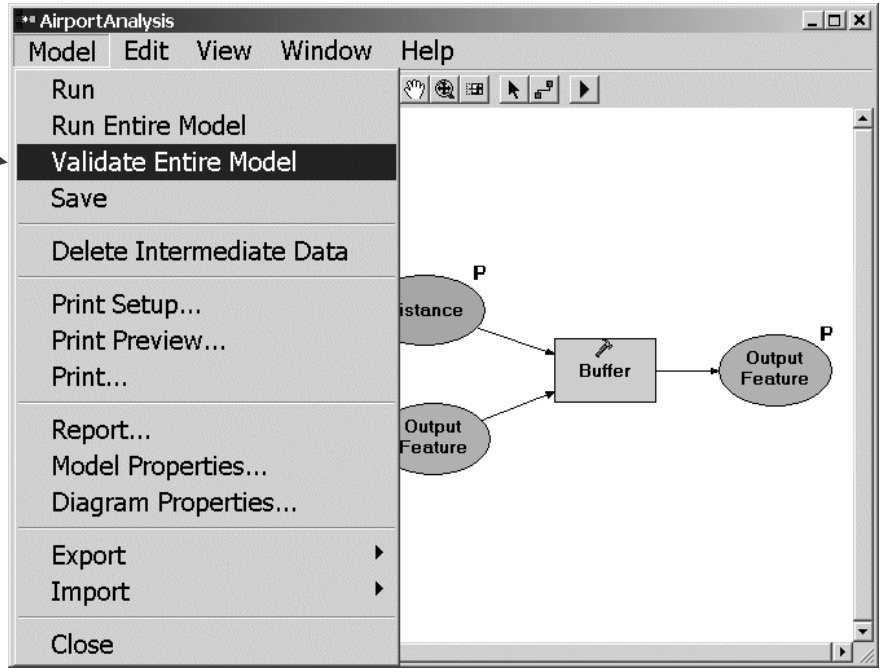
Three states of model elements

- Not ready to run: Parameters not defined
- Ready to run: All elements colored
- Already run: All elements colored and shadowed



Validating a model

- **Validating a model**
  - **Returns model elements to *ready-to-run* or *not ready-to-run* states**
  - **Validate to determine if model is valid an ready-to-run**



- **What causes a model to be invalid?**
  - **Parameter values no longer valid (referencing non-existent data)**
  - **Tools referenced no longer exist**
  - **COM tool inside the model is unregistered**

How to execute models

- **Models can be executed in 2 ways**

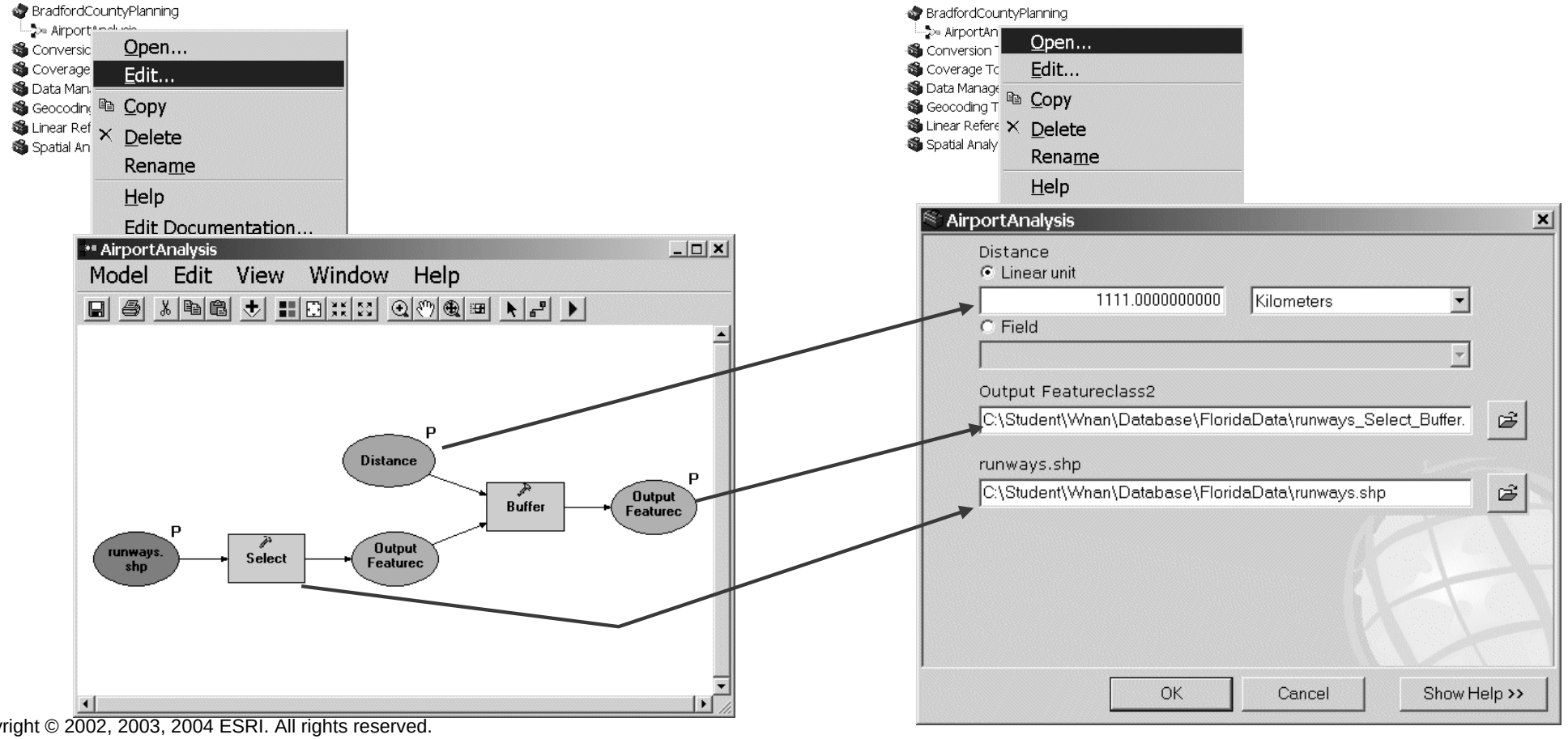
  1) **Directly from the ModelBuilder Window**
     - **Run entire model** ***Or***
     - **Can optionally execute one process at a time**
  2) **Execute from Toolbox as a tool dialog**

ModelBuilder

Tool Dialog

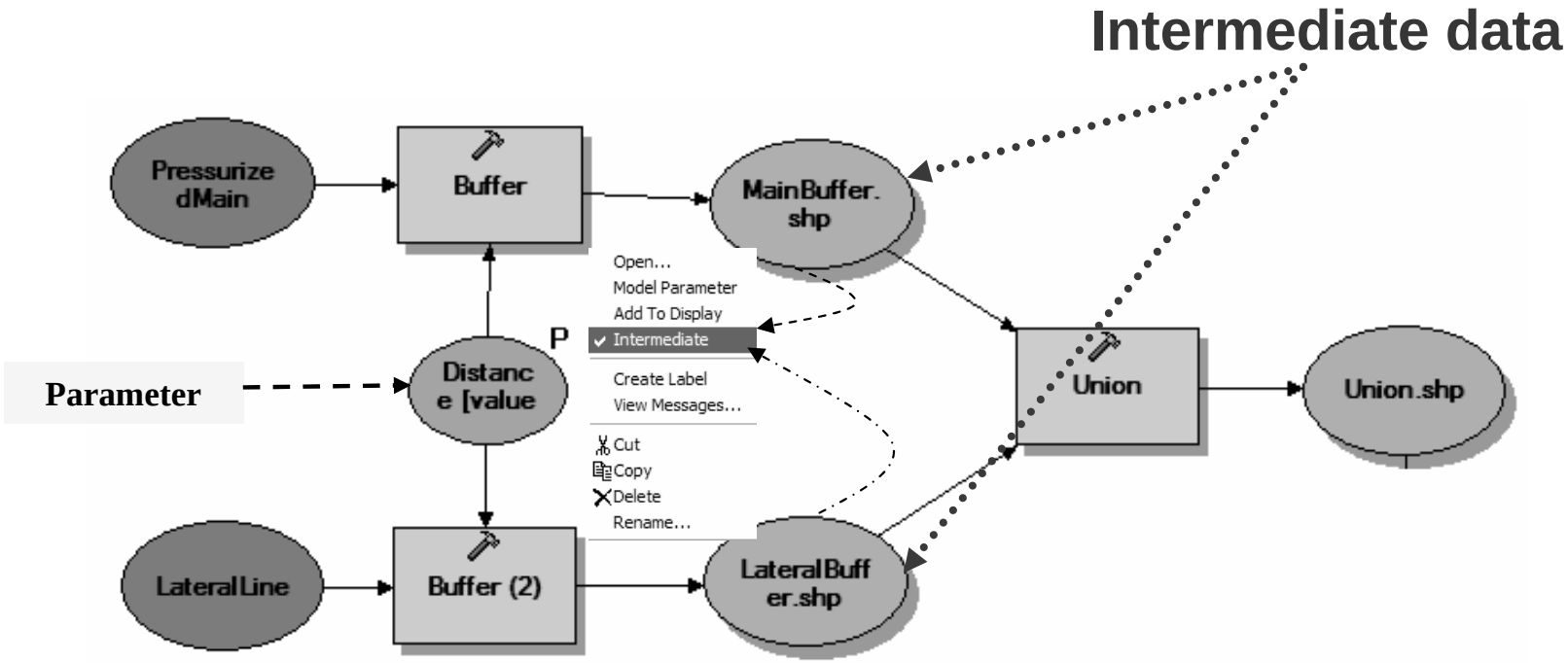
Running models with parameters

- Parameters include Input/Output data values and variables for a tool
- Right-click model element and choose to create parameter
- Parameters are used to run models as a dialog
  - All parameters created in a model appear as a tool dialog



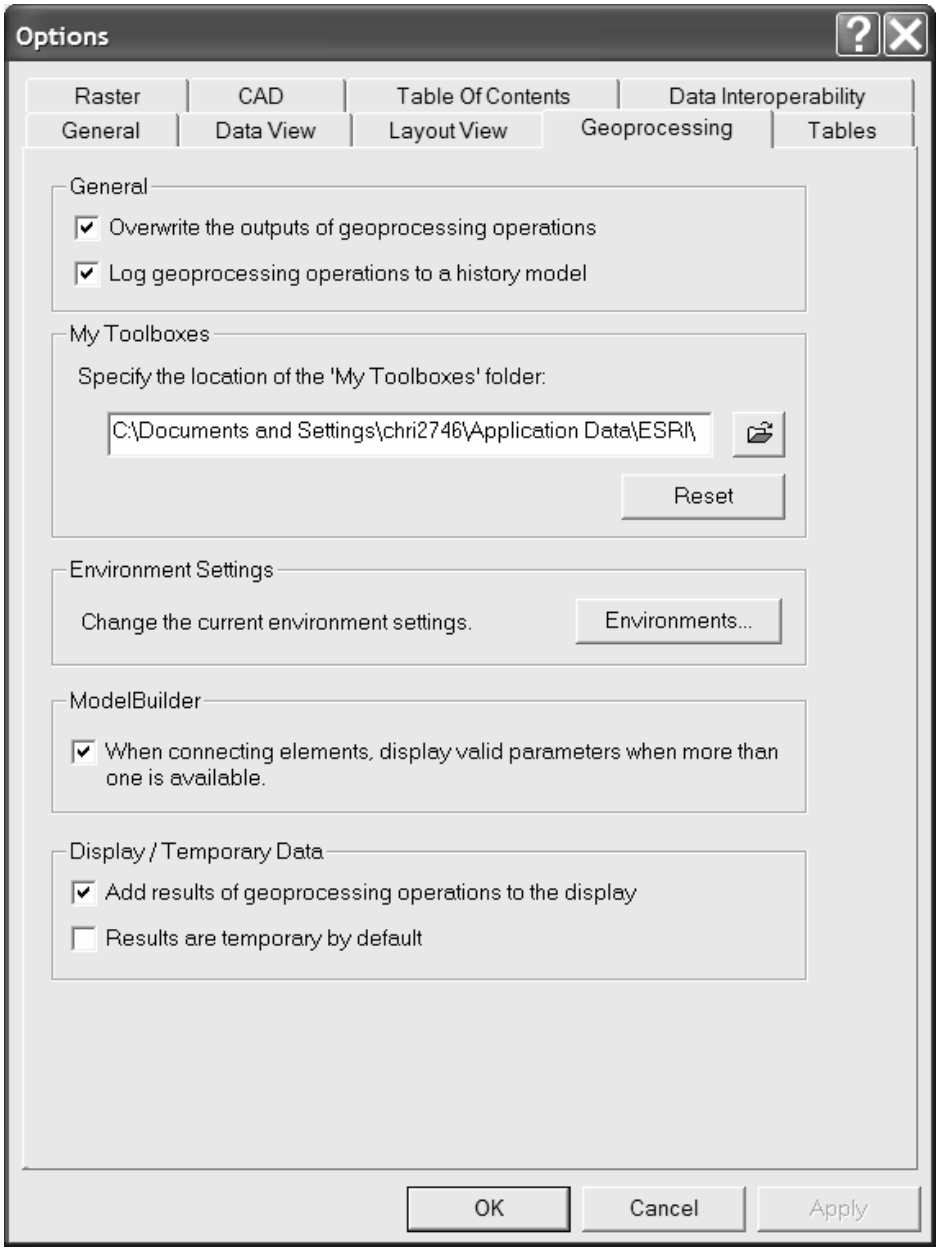
Intermediate Data

- You determine which derived data elements are intermediate
  - Right-click model element and choose ‘Intermediate’
- Intermediate Data is deleted after model executes
  - Saves disk space



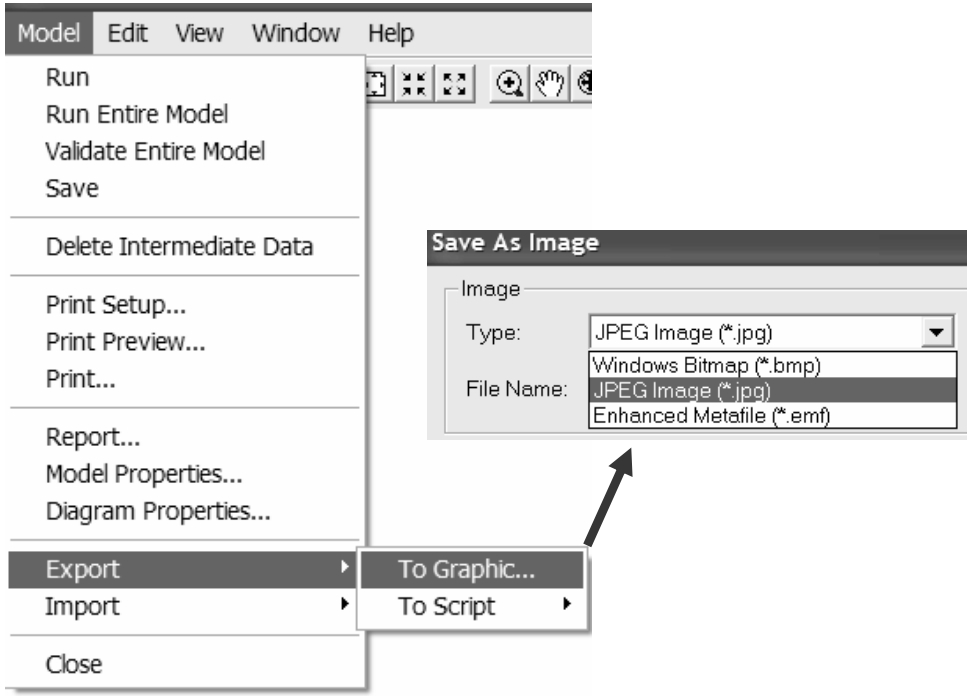
Geoprocessing options

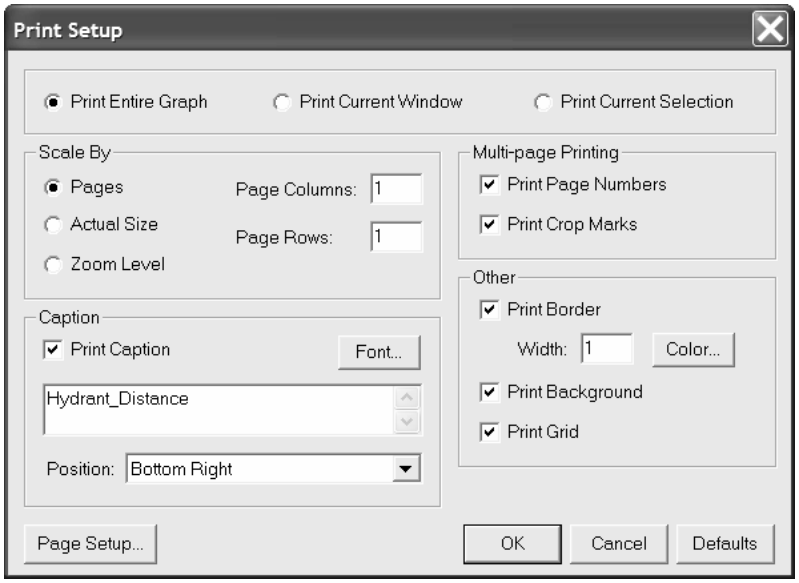
- **ArcMap & ArcCatalog**
  - **Tools → Options → Geoprocessing Tab**
- **Options:**
  - **Overwrite outputs of geoprocessing**
  - **Log geoprocessing to a history model**
  - **Specify your default toolbox location**
  - **Display valid parameters when connecting ModelBuilder elements**
  - **Add geoprocessing outputs to ArcMap display**
  - **Make geoprocessing results temporary**



Printing & Exporting models

- **Export to a graphic**
  - **Export as .bmp, .jpg, .emf**
  - **Useful for placing in map layouts**
- **Printing models**
  - **Add borders, captions, page numbers**

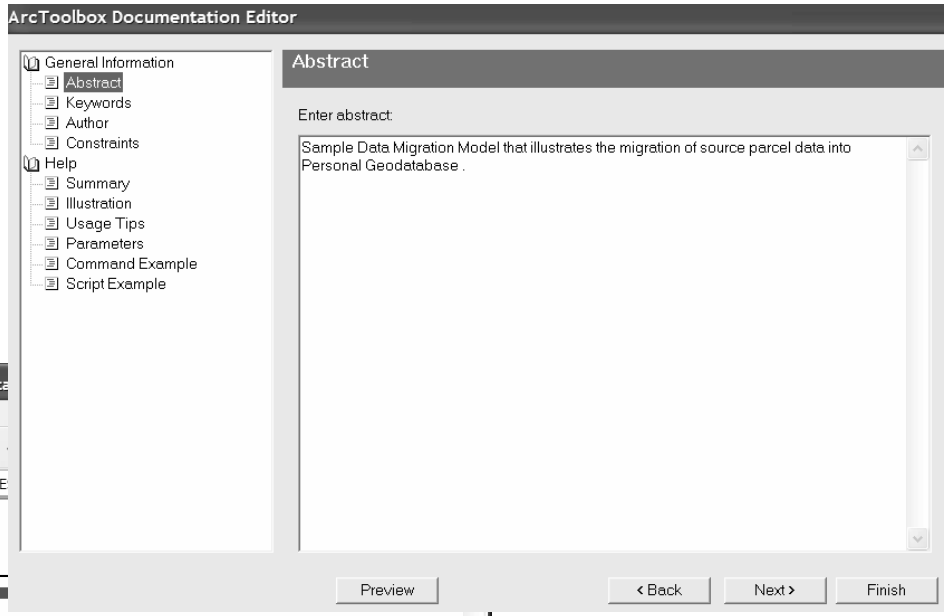


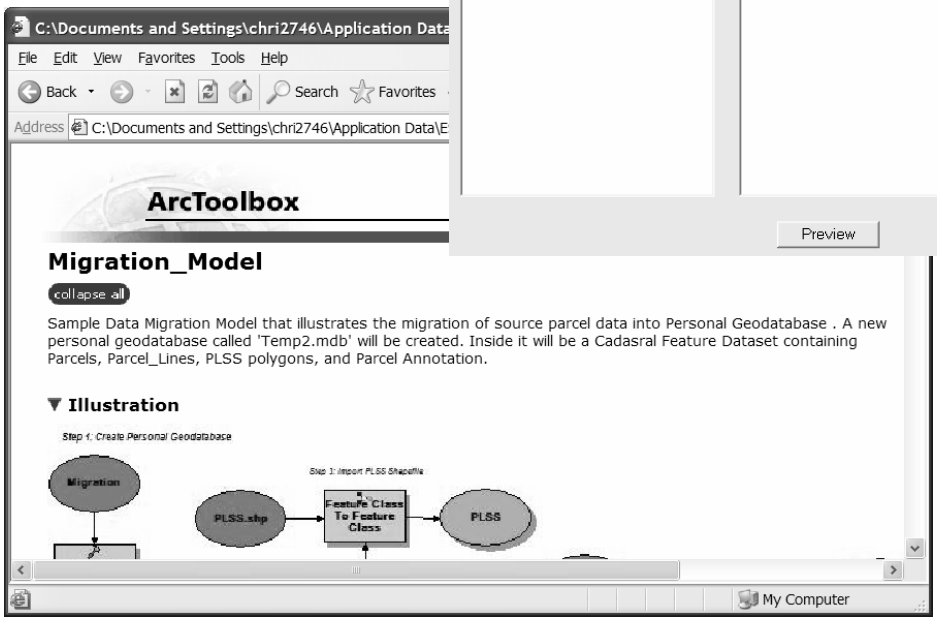


Tool documentation

Documentation Editor

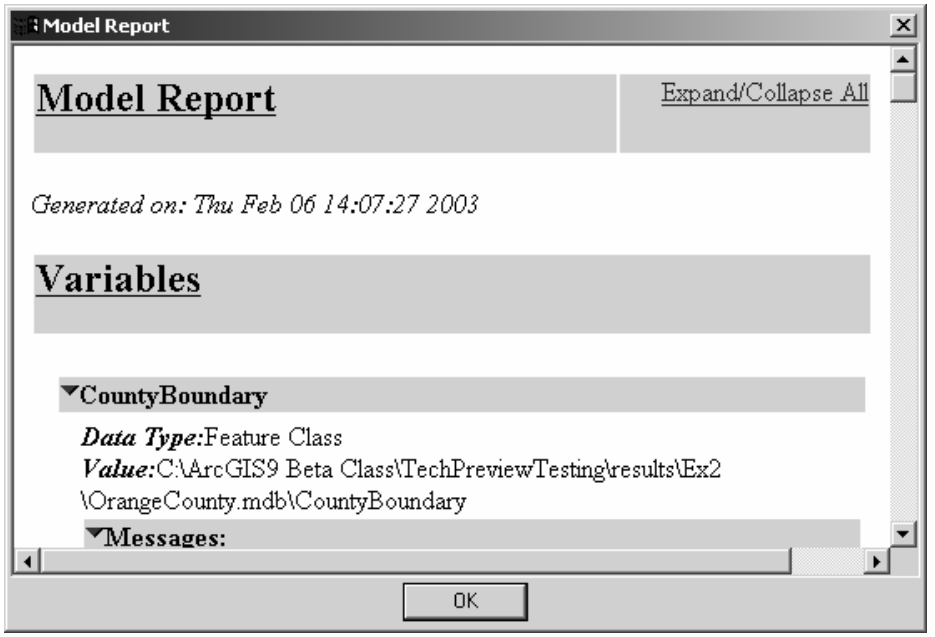
- **Ability to edit documentation for tools and toolboxes**
- **Ability to create help menu**
  - **Documentation built into ArcGIS online help system**
- **Ability to create metadata**
  - **Useful for Metadata server**





Model Report

- XML document available for viewing or creation
- Useful for historical view of processes executed
- Contains information on:
  - Variables
  - Processes
  - Messages
  - Time and date of execution



Saving and sharing models

- **Models can be saved to a stand-alone Toolbox (.tbx) or to a toolbox within a Geodatabase (.mdb)**
  - **Exchange the .tbx or .mdb file**
- **Option to export the model as a script and share the script**
  - **Python, VBScript, or Jscript**

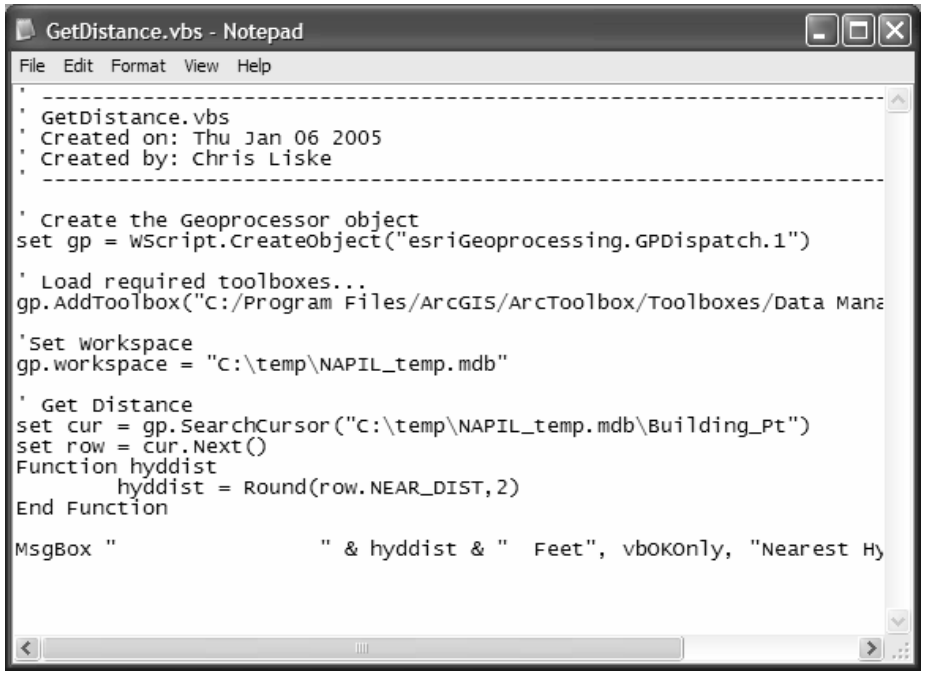
***** Note: Set model parameters if the tool will be used with different data***

Demonstration:
ArcToolbox & ModelBuilder

Scripting and the Geoprocessor

Why write scripts for geoprocessing?

- Similar advantages that models have
  - Efficiently execute series of different tasks
  - Easy to read and document
  - Easy to share
- Perform batch operations
- Use logic to control tool execution
  - Branching and Looping
  - Delayed Processing
- Interface with other systems
  - Can be run outside of ArcGIS
- Self contained (single file)
- Run any time
- Familiar environment for AML and Avenue users
  - Users don't have to learn a proprietary language



Where to write scripts?

- **Choice of scripting languages: Any language with a COM interpreter**

VBScript

Based on VB: Simpler

Popular Web language

JScript

Syntax similar to C/Java

Works in many Web browsers

Python

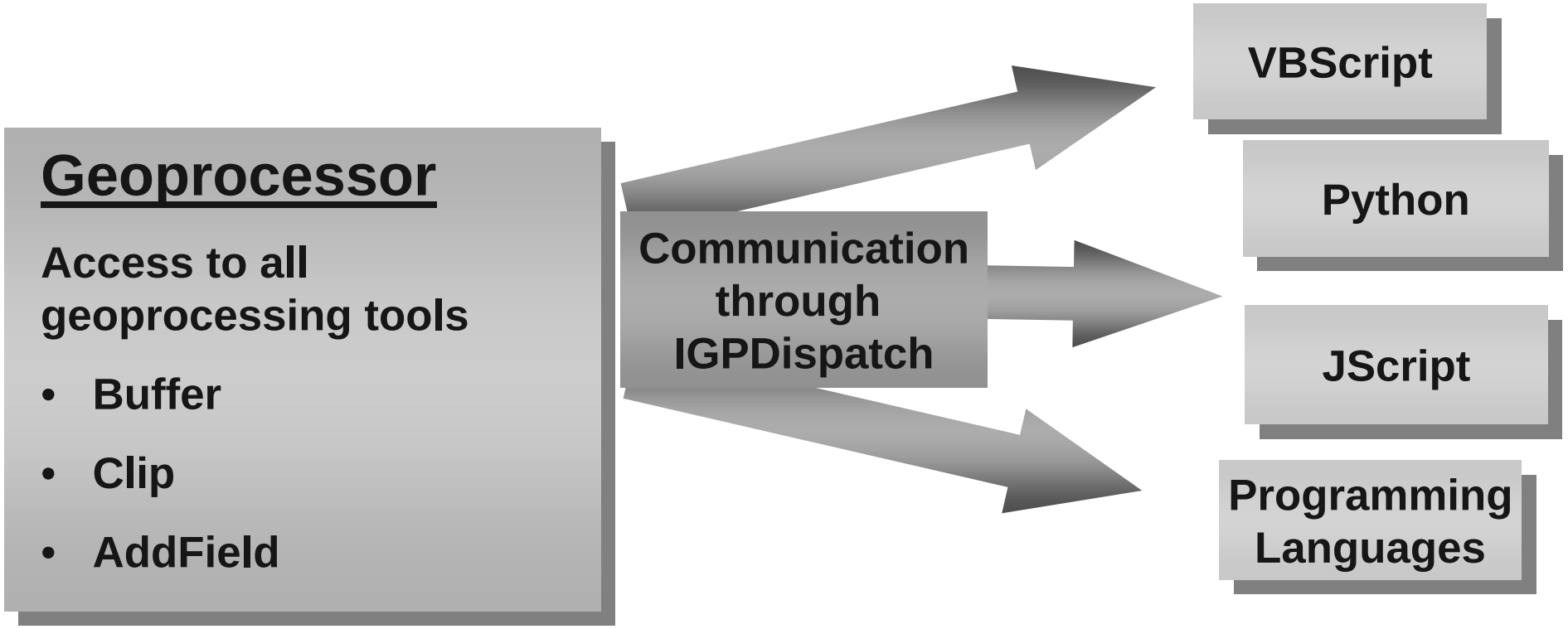
Flexible, powerful

Easy to learn

Perl and other scripting languages

GpDispatch

- An ArcObject accessible in all COM languages/scripts
- Also referred to as the Geoprocessor
- Exposes all geoprocessing functionality
  - Object is late-bound (no code completion)



Writing scripts

- **Import Geoprocessor object**
  - ***Uses native Python support , scripts not dependant on Windows OS at 9.2***

```
import arcgisscripting
```

- **Instantiate the Geoprocessor object**

```
gp = arcgisscripting.create()
```

- **Set properties (e.g., where processing will occur?)**

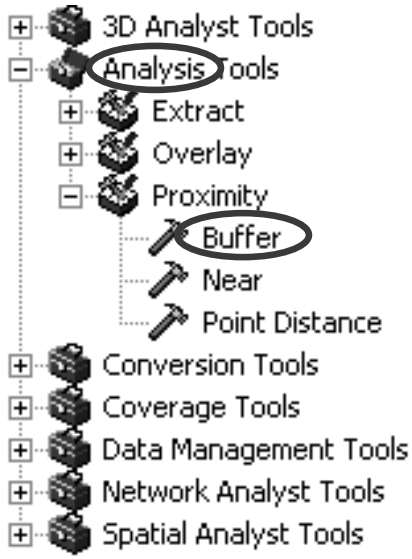
```
gp.workspace = "c:\\Florida.mdb"
```

- **Comment code**

```
# Buffer roads by 100 meters
```

- **Run tools**

```
gp.Buffer_analysis("roads", "rdbuf100", "100")
```



Constructing Python statements

- **Use the Geoprocessor with object-oriented programming**
- **Call tools with:**

Geoprocessor.ToolName (parameter1, .., parameterN)

```
gp.CreateFolder("c:\\Florida", "Scripts")
gp.Clean("Europe", "#", "#", "#", "POLY")
```

Referencing data in Python

- Use \\ or / for path names
  - Python is based upon the C programming language
- Defined by double (") or single (') quotes

```
gp.workspace = "c:\\Florida"
gp.CreateFolder('c:\\Florida', 'Scripts')
```

- Concatenate strings

```
outputfc = gp.workspace + "\\" + inputfc
```

Referencing tools and toolboxes

- Only system toolboxes available by default from Geoprocessor
  - Analysis, coverage, data management, etc.
  - Use toolbox alias if tools have the same name

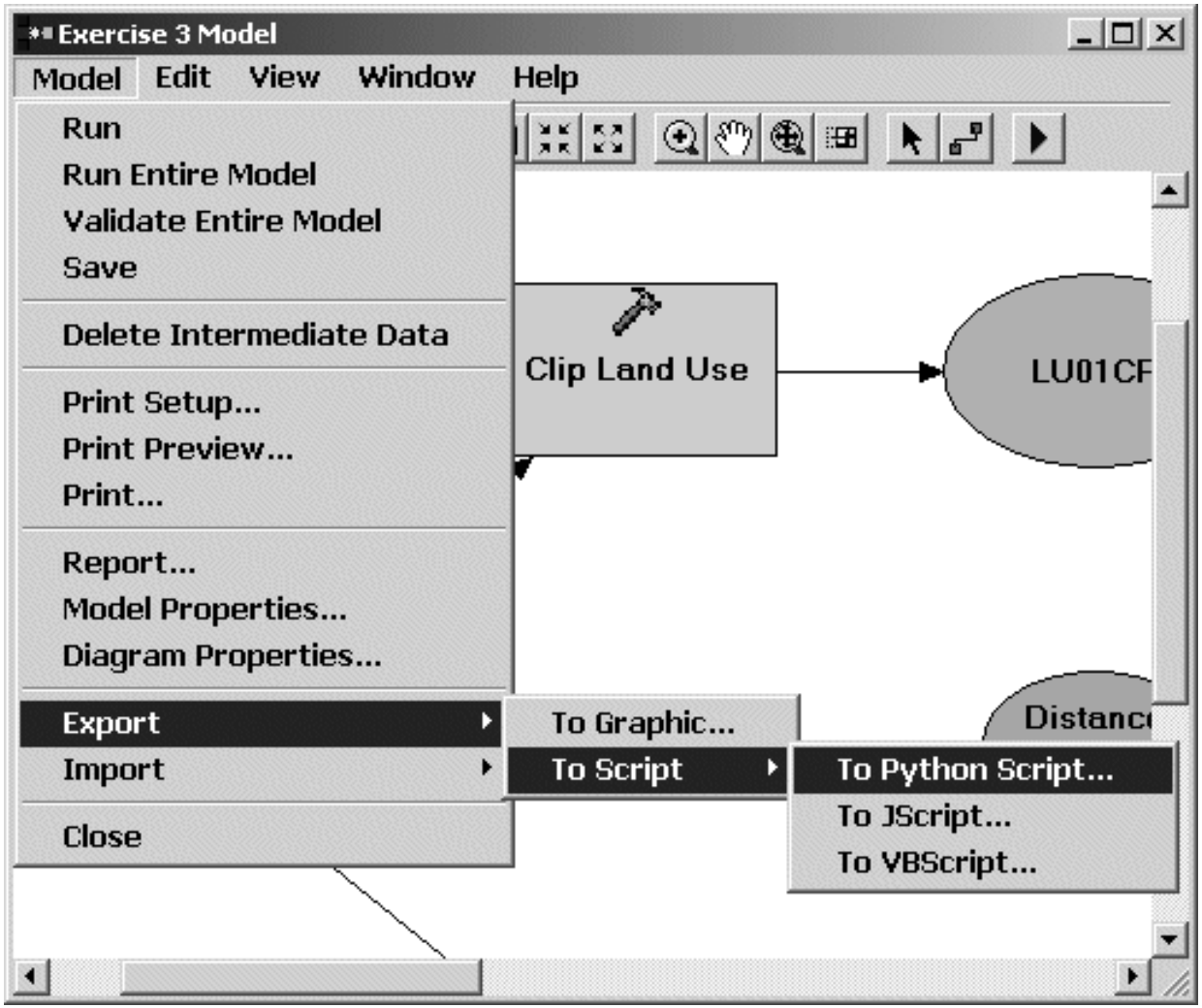
```
gp.Buffer_analysis(roads, Out_Buff, "10 Meters", "FULL")
```

- Add your own toolbox

```
gp.AddToolbox("c:\\MyProject\\CrossCountryMobility.tbx")
gp.BestPath("start.shp", "destination.shp", "results.shp")
```

Exporting a model to a script

- **Easy way to start writing scripts**

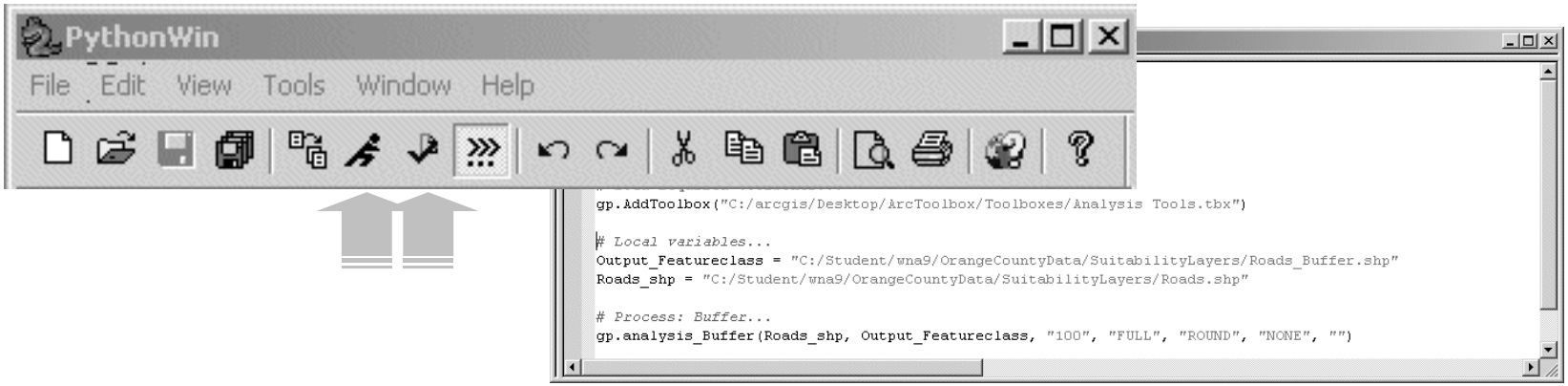


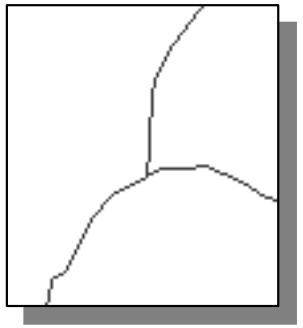
Example script

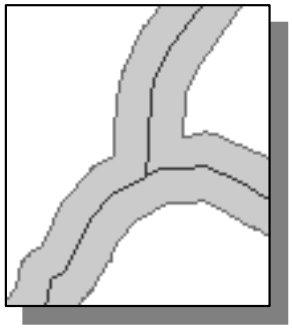
```
# Import system modules
import sys, string, os, arcgisscripting

# Create the Geoprocessor object
gp = arcgisscripting.create()

roads = "C:/Data/Florida/ROADS.shp"
Out_Buff = "C:/Data/Florida.mdb/ROADS_Buffer"
citylimit = "C:/Data/Florida/citylimit.shp"
Out_Clip = "C:/Data/Florida.mdb/BUFF_Clip"

# Process: Buffer...
gp.Buffer_analysis(roads, Out_Buff, "10", "FULL", "ROUND")
# Process: Clip...
gp.Clip_analysis(Out_Buff, citylimit, Out_Clip)
```

Running scripts in different environments

- Check syntax and run in PythonWin
  - ArcGIS applications can be closed







- Run at DOS prompt

```
Python C:\PythonScripts\RoadBuffer.py
```

- Create a tool
  - Add a script to a toolbox
  - Open command runs script tool
  - Edit command opens script source in text editor

Distributing scripts

- As Python source file .py
- As script tools in toolboxes
  - User creates script tool for parameters
  - Share the .py file and toolbox

Using the Command Line Window

Command Line window

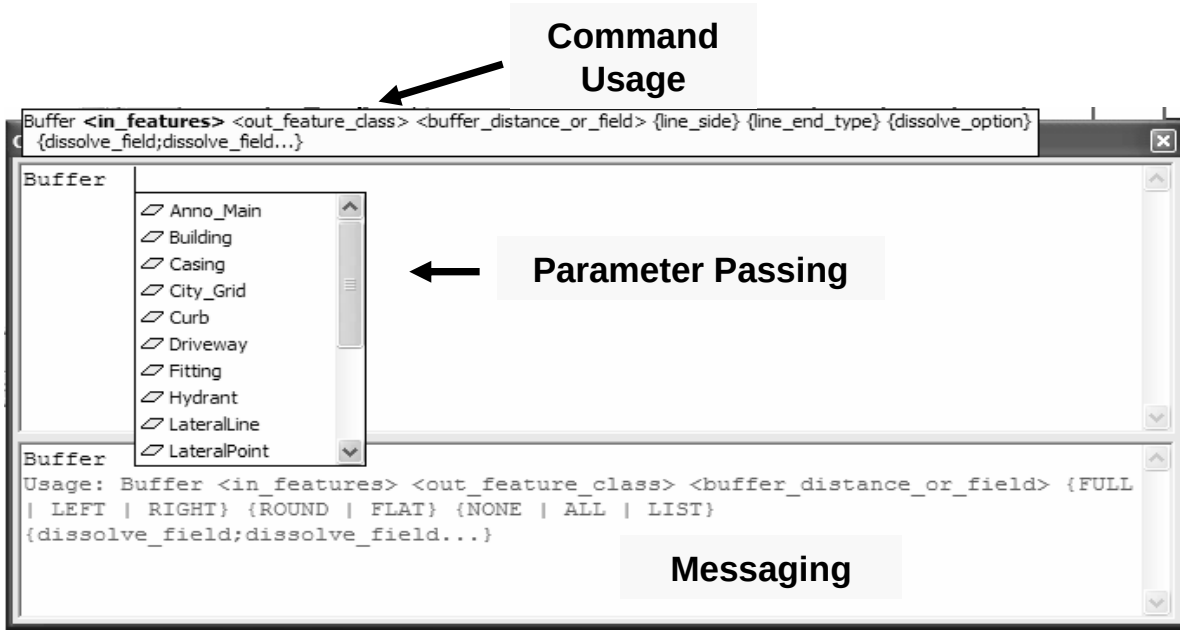
- **Available in all ArcGIS applications**
- **Similar interface to ArcInfo Workstation but with more intelligence**
  - **Auto Completion, Help, and intuitive messaging**
  - **Ability to save environment settings and create variables**
  - **Parameters provided in a dropdown list**
  - **Commands and arguments may be different**

ArcInfo Workstation

ArcGIS 9x

Why use command line?

- Execute models, scripts, and tools quickly
- Shortcut to tools in ArcToolbox
- Saves time if you're familiar with geoprocessing tools
  - Allows you to obtain quick view of all tools available



Syntax notation

- **Similar to ArcInfo Workstation**

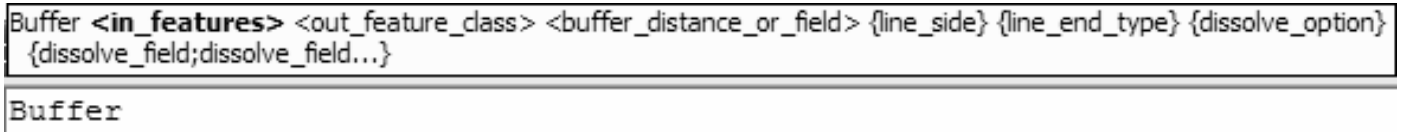
```
ToolName <Argument> {Argument} {KEYWORD|KEYWORD}
```

Required *Optional* *Optional Keywords*

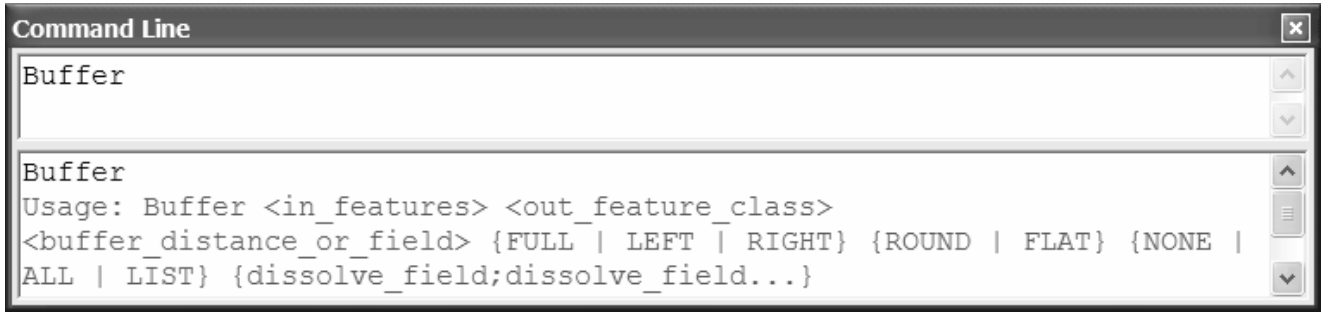
- **First optional keyword will be taken as default**
- **Use # to skip arguments**
- **Allows multi-line commands**
  - **Shift + Enter**
  - **Press Enter to execute all tools in the Command Line Window**

Passing in parameters

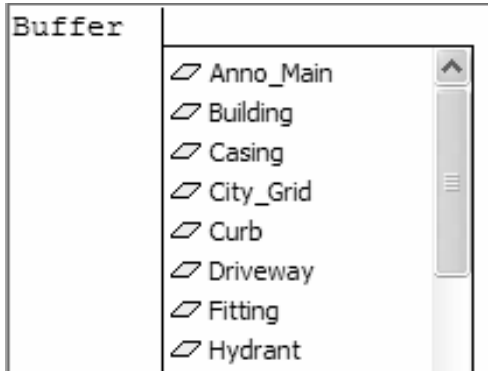
- **Usage appears showing all parameters**



- **Type in name of command and press Enter**
  - **Usage appears in Command Line window**

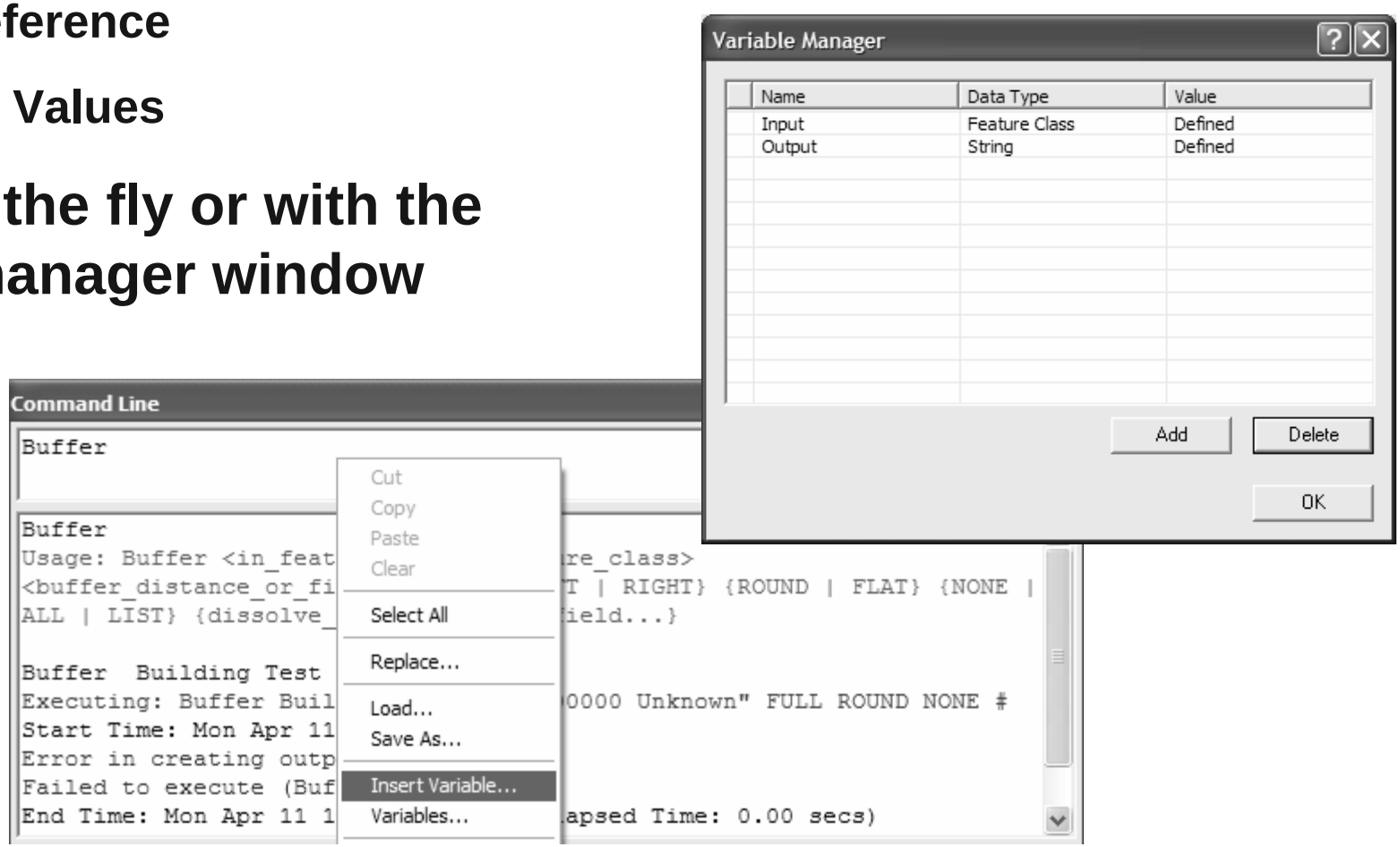


- **Dropdown list shows available layers in ArcMap TOC**



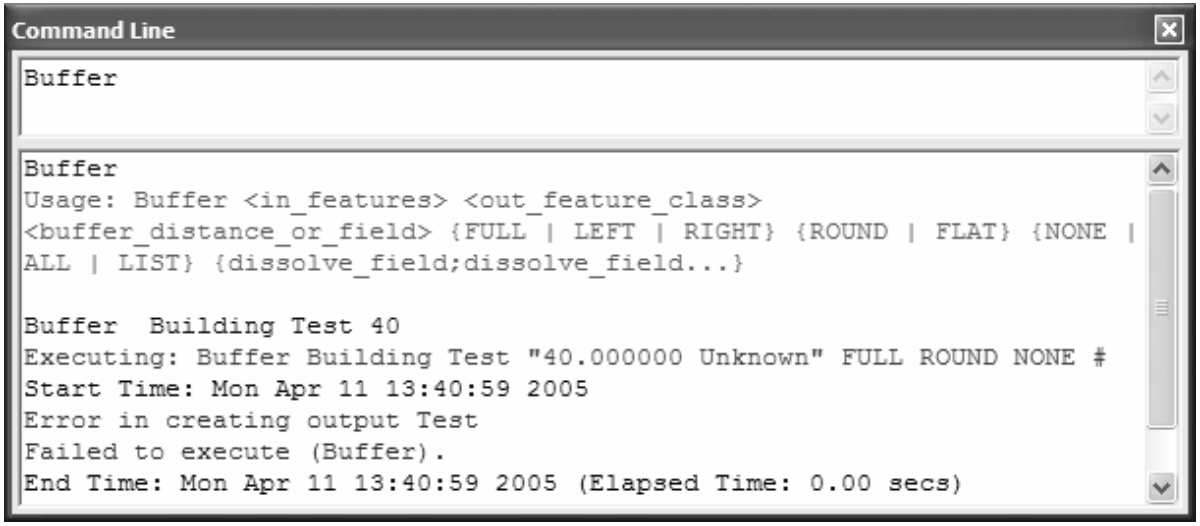
Using Variables in Command line

- **Make variables for parameters**
  - **Data and workspaces**
  - **Expressions**
  - **Spatial reference**
  - **Strings & Values**
- **Create on the fly or with the variable manager window**



Messages

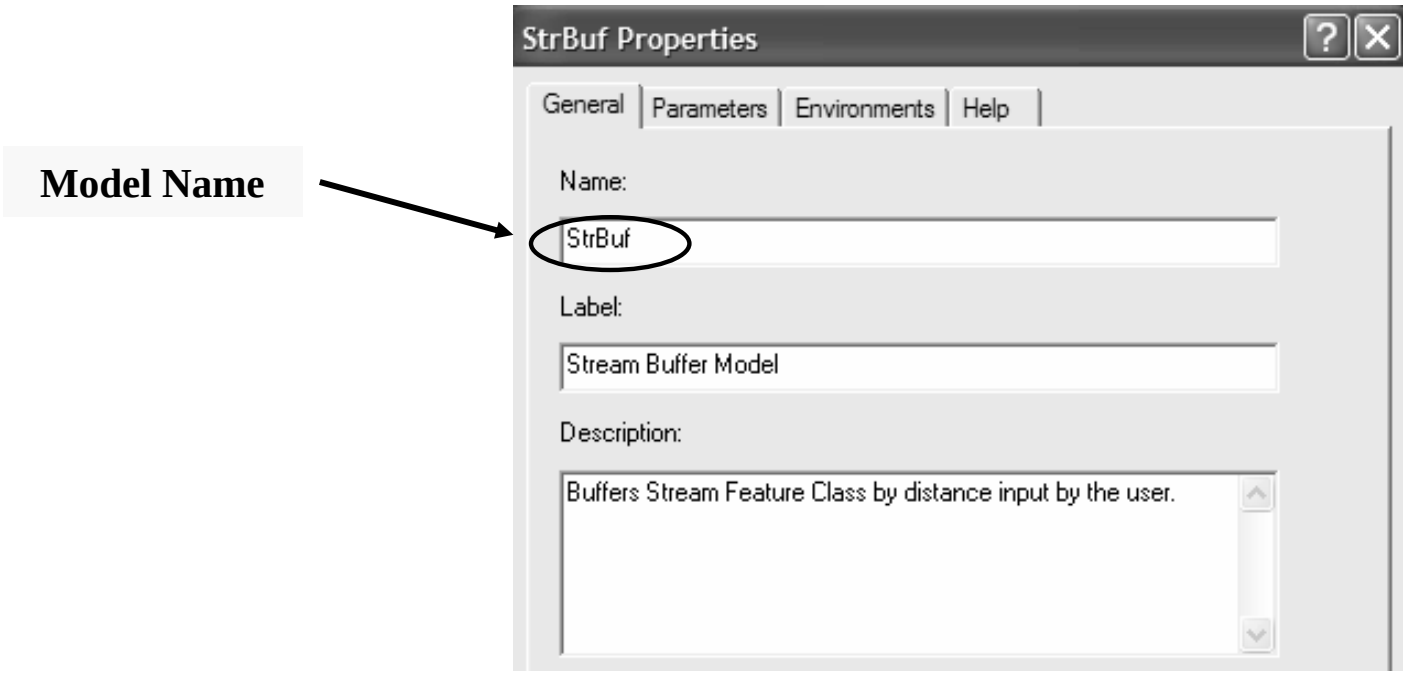
- **Successful completion of a tool**
  - **Includes Start and End Time**
- **Commands (blue font)**
- **Errors (red font)**
  - **Indicates a tool did not execute**
- **Warnings (green font)**
  - **Indicates that a step may have been skipped; gives usage**



Executing a model from command line

2 Rules

- Toolbox containing model must be added in ArcToolbox
- Execute model by calling model name, not the label



Editing and re-executing existing processes

- **Allows re-execution of previous commands** ***Or***
- **Allows to edit parameters of previous commands before re-executing**
  - **If source of layers changes**
  - **Change a parameter (buffer distance)**
  - **Edits are made (new features added)**
  - **Selection changes, etc…**

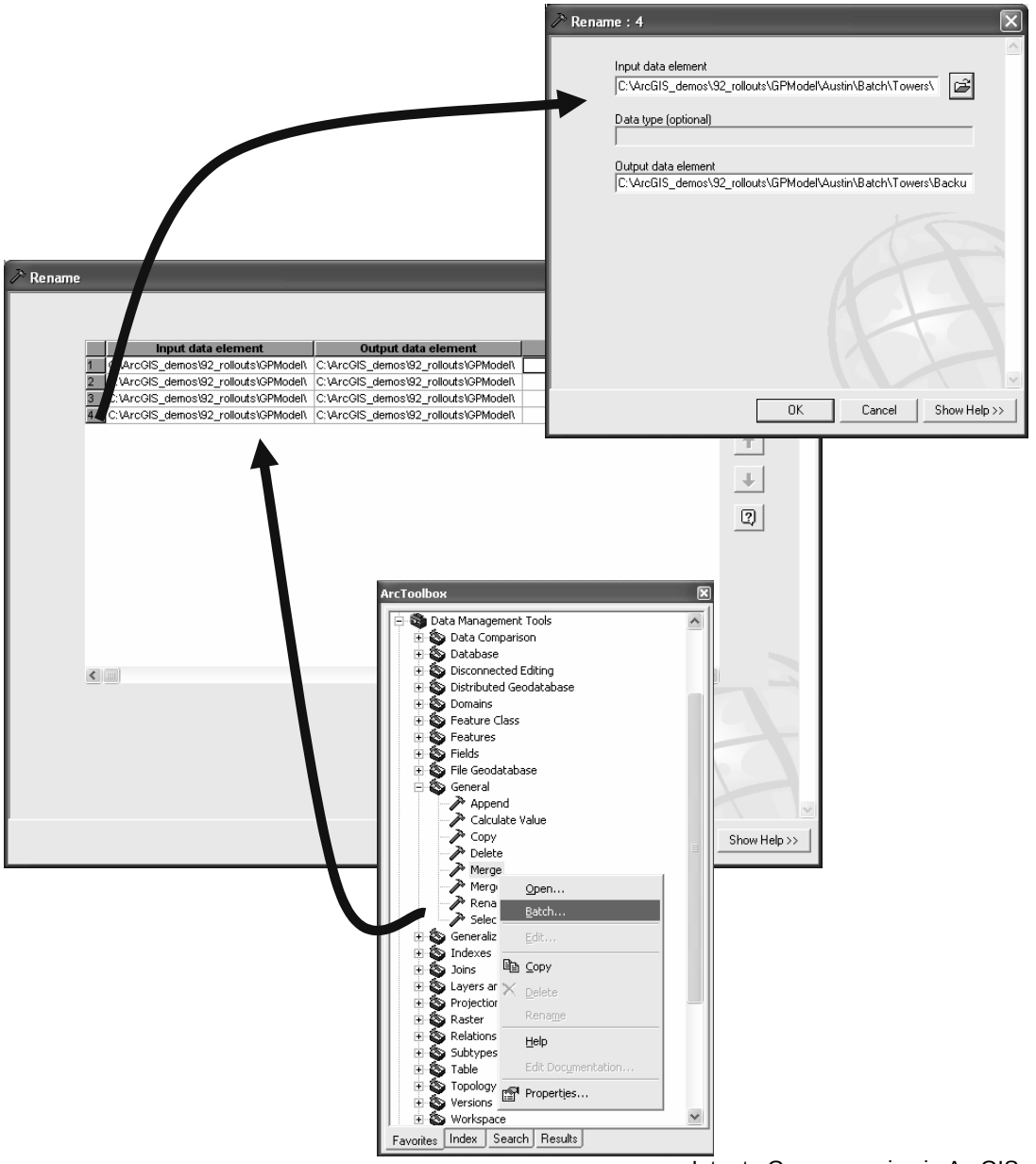
***** Can also use up and down arrows on keyboard for previous commands***

Demonstration:
Scripting & Command Line

Advanced Geoprocessing and Modeling

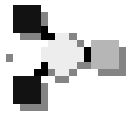
Batch Mode

- New at 9.2
- Run the same tool many times with different inputs, parameters, or outputs
- Available for all tools, models, or scripts
- Right click Tool > Batch
- No script necessary



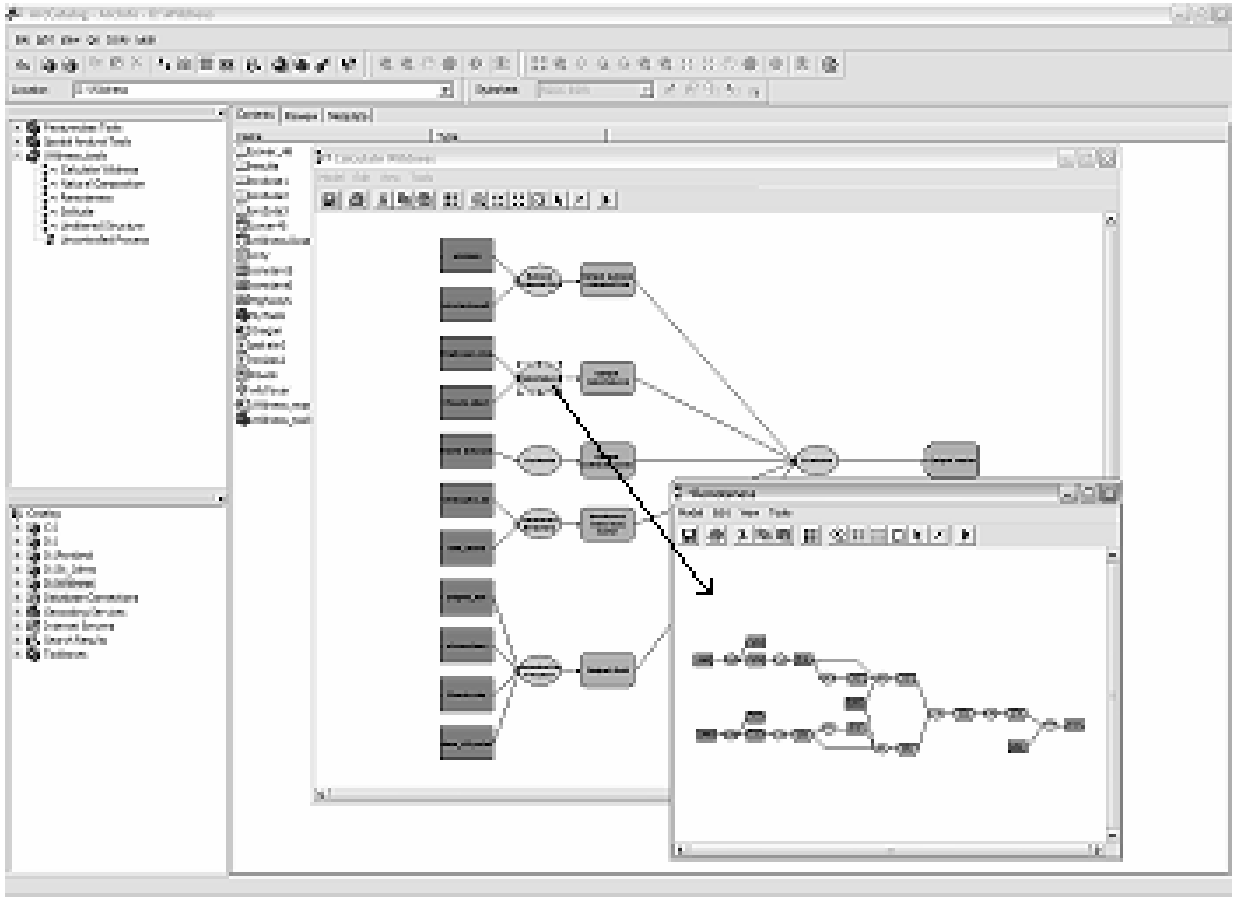
Many inputs to Models

- Models can carry out actions beyond what is within the toolbox
  - Run a previous model as a tool
  - Run a script as a tool
  - Use an .exe or external function as part of a model



Running a model in another model

- **Drag and drop model from toolbox like any other tool**
- **If output from a previous model is a parameter, it will be exposed in a larger model**



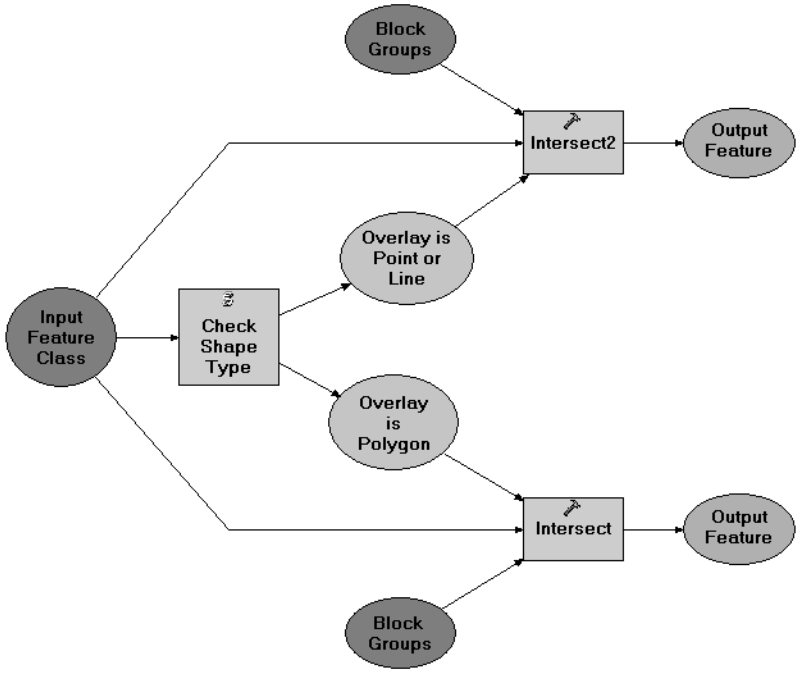
Using a script

- Drag and drop a script from a toolbox
  - Create parameters in script to add connections between tool
- Attach an .exe or AML to a script

Precondition property

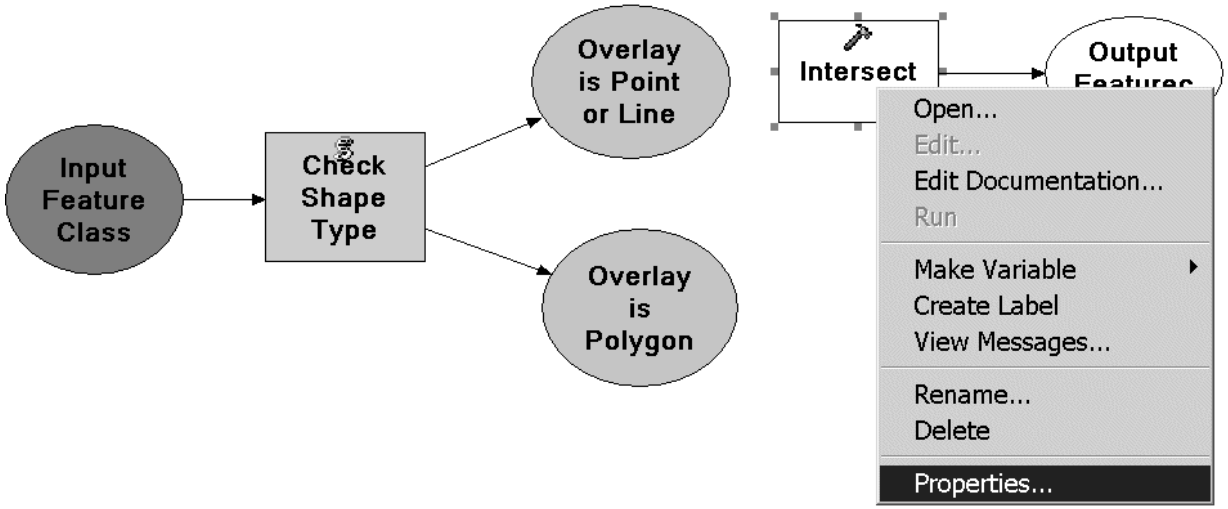
- Used to control flow of model execution

- Used to branch within a model

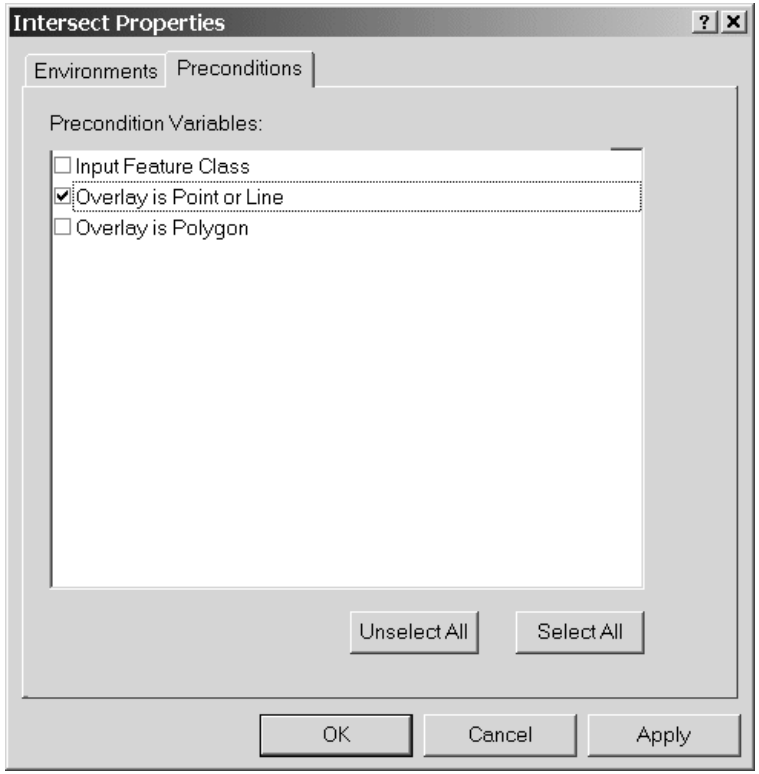


Applying a precondition property

- **Set for tools accepting parameters from a script**



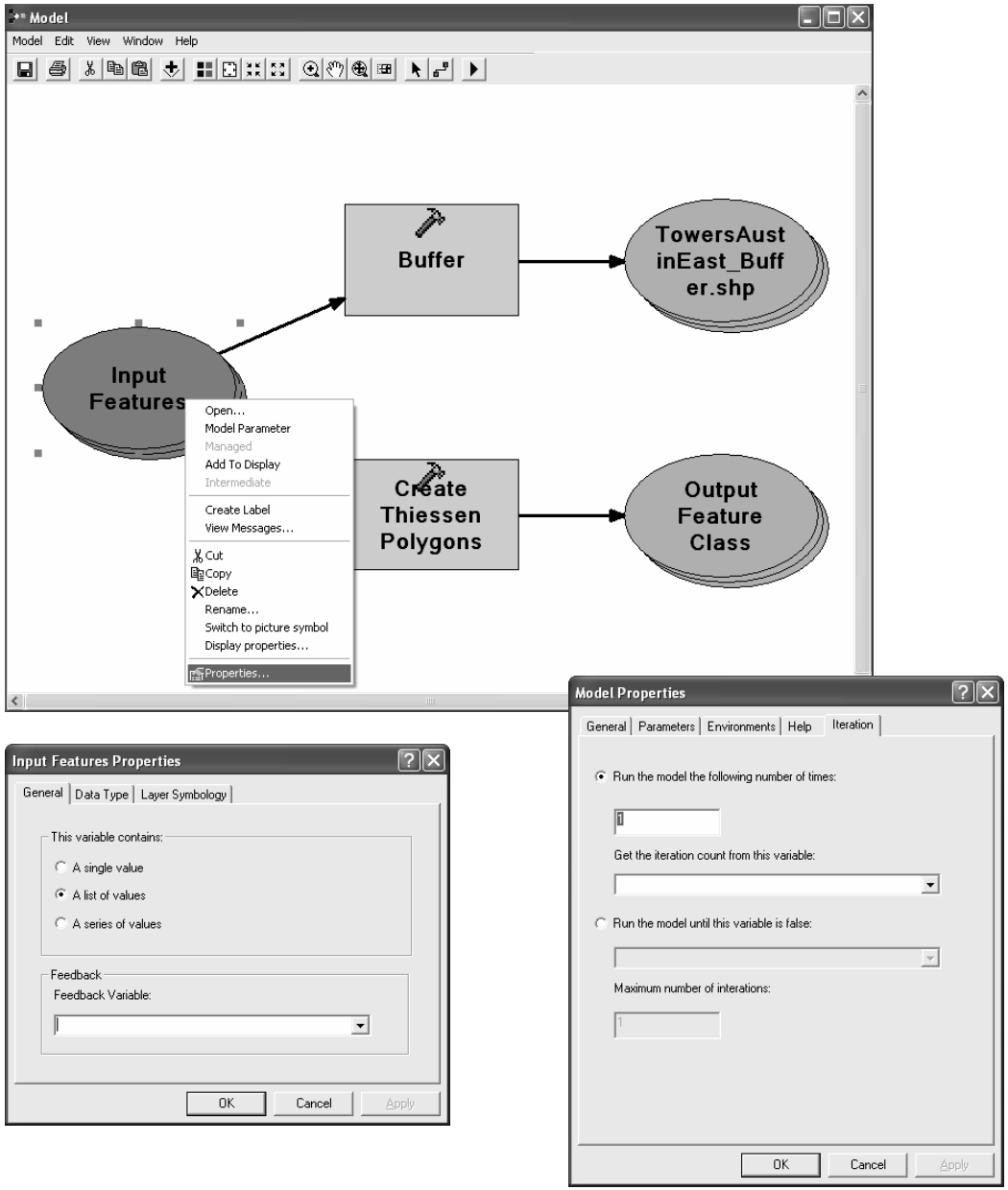
1. Apply Precondition to next tool



2. Select condition from script output

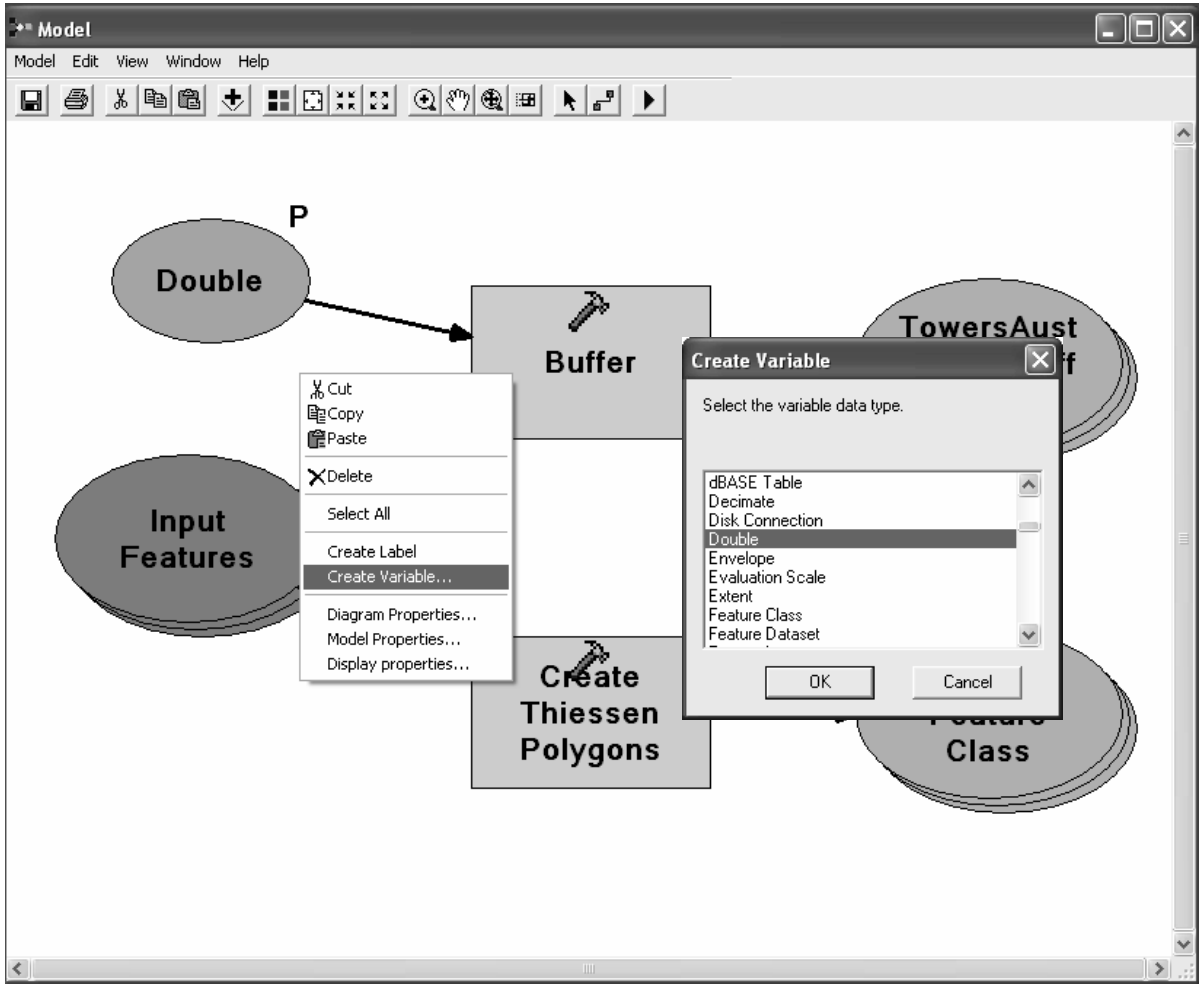
Looping in a model

- New at 9.2
- Applied to whole model or specific processes
- Supports higher logic
  - Lists
  - Series
  - Boolean conditions
  - Counts
  - Feedback



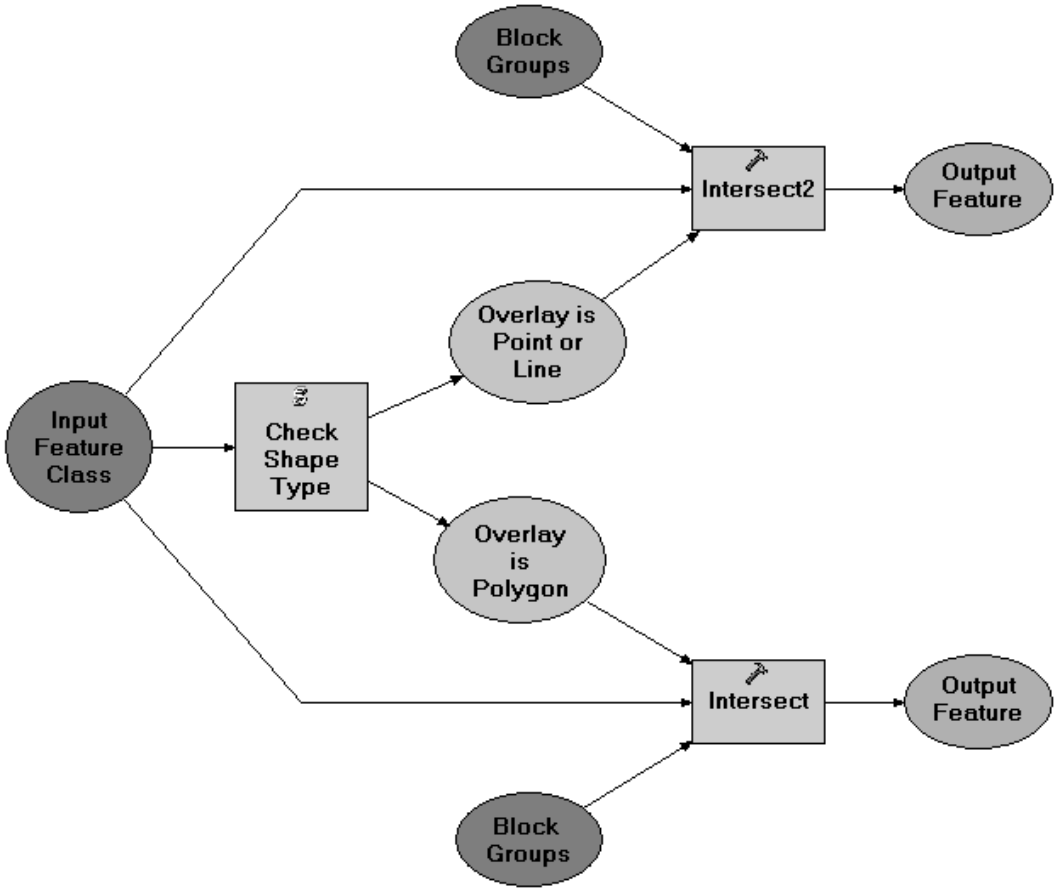
Variables in models

- New at 9.2
- Place holder for numbers, strings, or other inputs to processes
- “Any value” option
- Can be set by other process or by user



Branching in a model

- **Precondition property from a script**



Steps for branching within a model

❶ Create a script that will set parameters as output-based upon if/then conditions

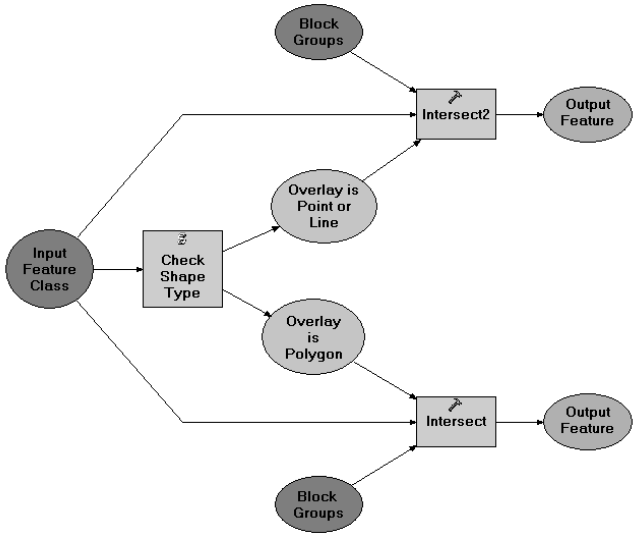
❷ Add the script to a toolbox, specifying output parameters that are in the script

❸ Apply precondition properties to tools to determine model flow

Writing a script to branch in a model

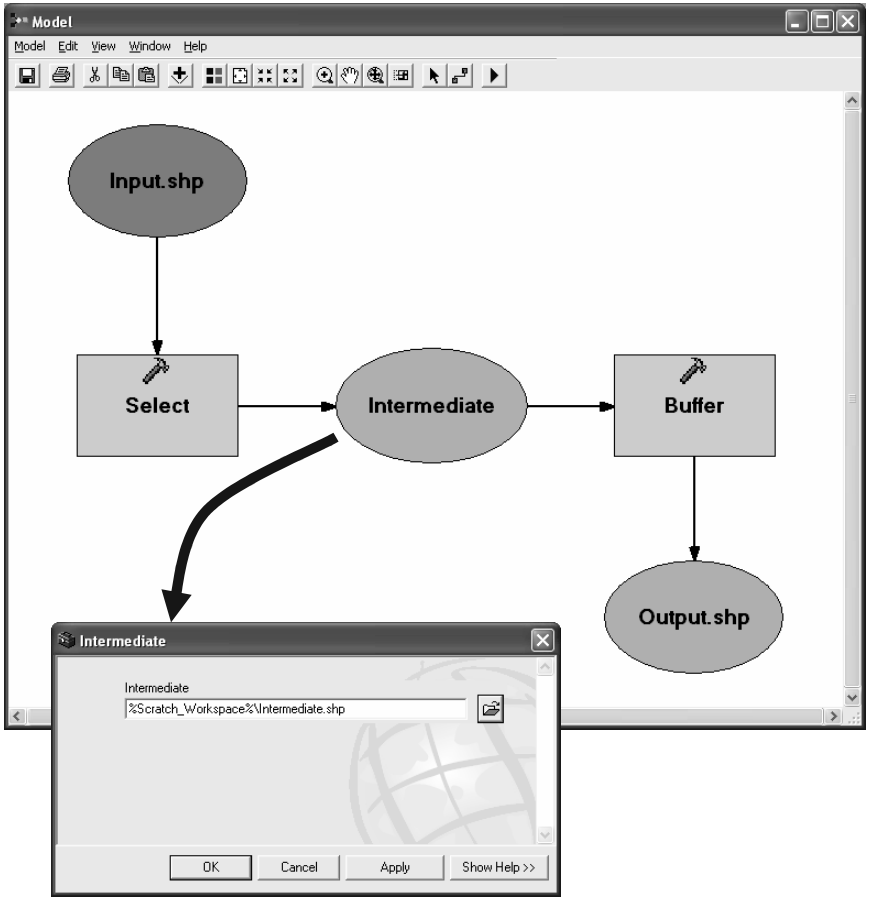
- Specify parameters to control which tools are executed in a model

```
inputfc = gp.GetParameterAsText(0) #SAME AS sys.argv[1]
dsc = gp.describe(inputfc)
if dsc.shapetype == "polygon":
    gp.SetParameterAsText(1, "True")
    gp.SetParameterAsText(2, "False")
else:
    gp.SetParameterAsText(1, "False")
    gp.SetParameterAsText(2, "True")
```



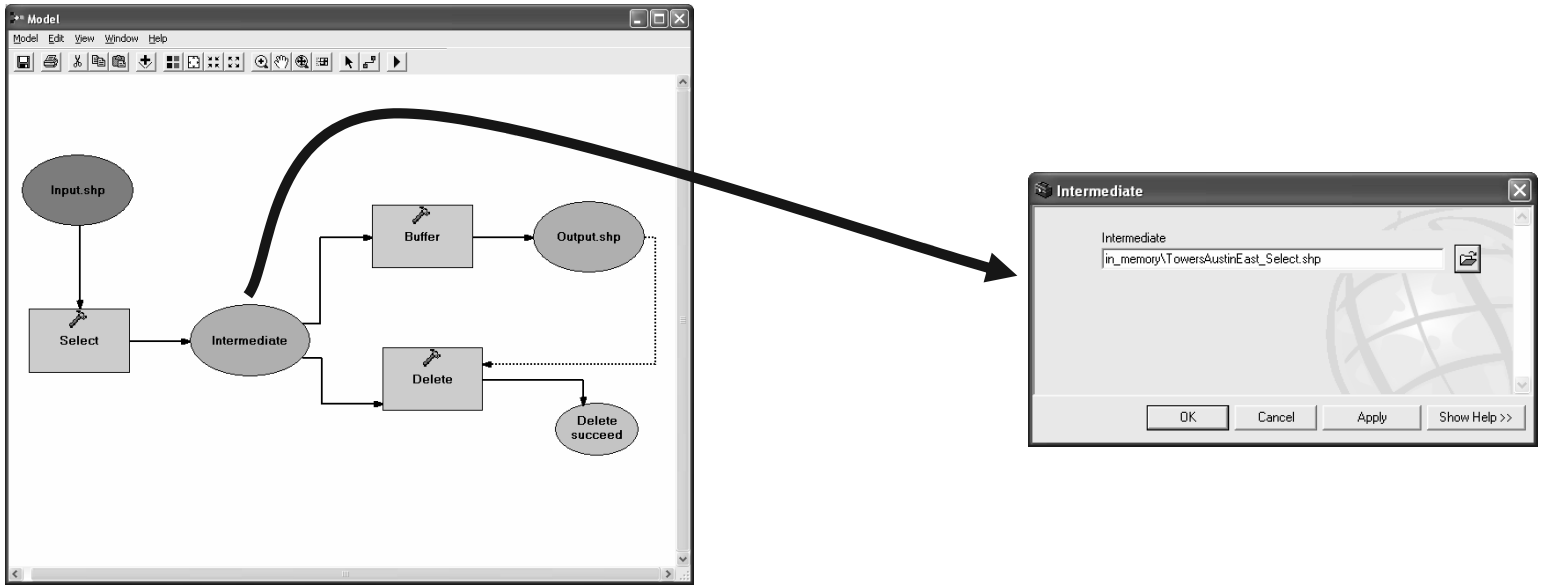
In-line variables

- New at 9.2
- Available in ModelBuilder and Command line
- Substitutes for another variable
- Enclose variable in %
  - %ScratchWorkspace%
- More portable models
- Two system variables
  - %n% - Number in loop, starts at 0
  - %i% - Number in series, starts at 0



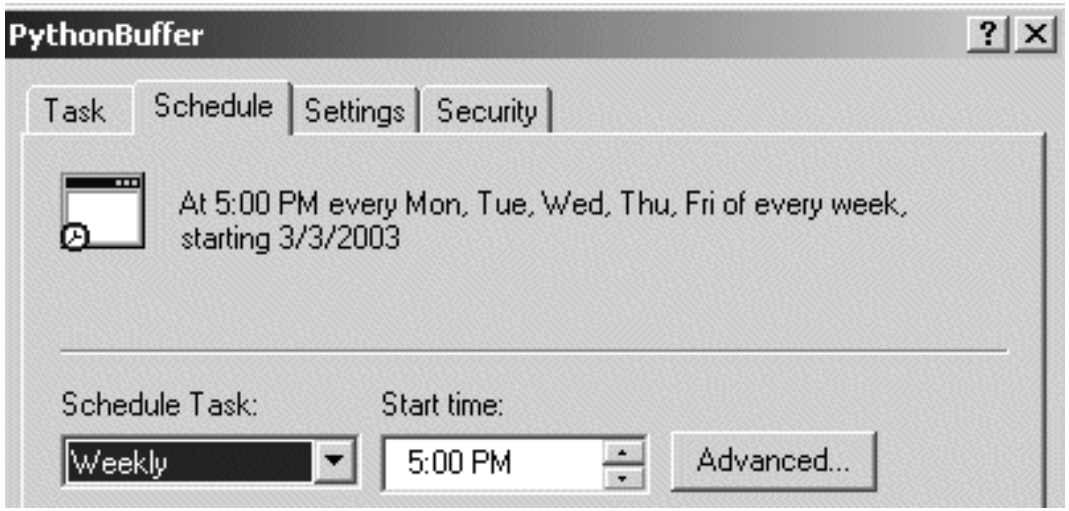
“In_memory” workspace

- New in 9.2
- Output written to memory, not to disk
- Much faster models
- Only Intermediate data
- Should use Delete tool to manage data



Run a script at a specific time

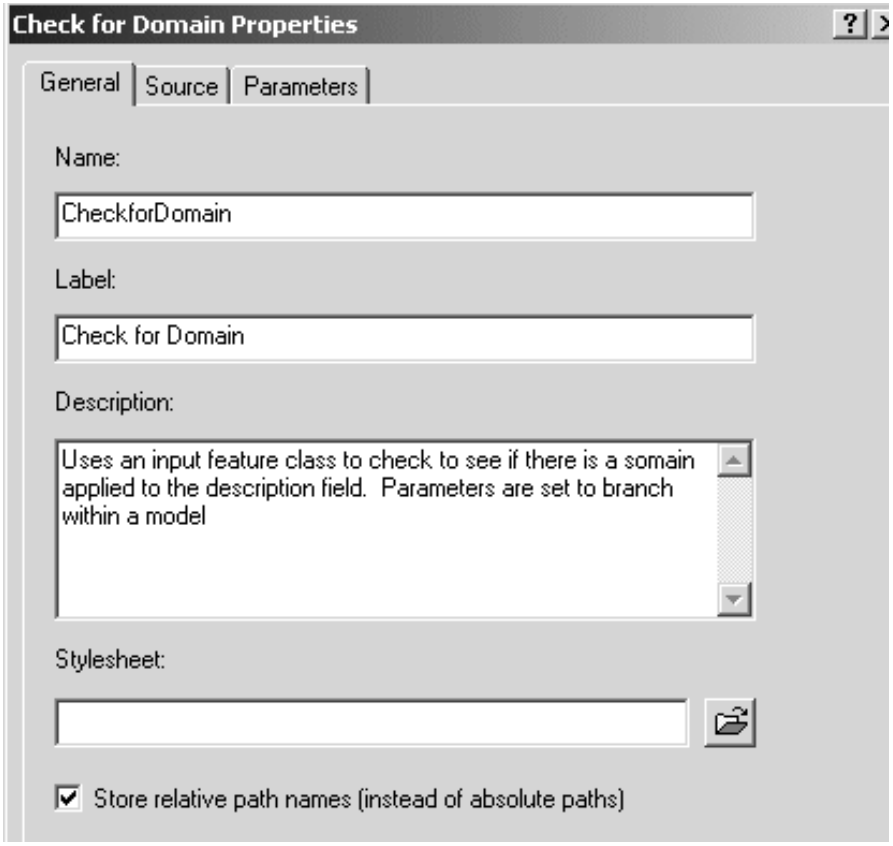
- **Windows scheduler**
  - **Start Menu > Settings > Control Panel > Scheduled Tasks > Add Scheduled Task**
    - **Browse to Script**
    - **Select Program you want Windows to run (ie. Python)**



http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=27553

Relative paths

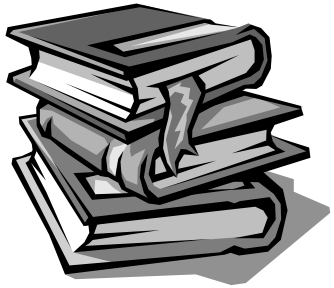
- Property of a model or a script
- Keeps track of all tools and data elements
  - Stores reference information to the source data



Store Relative Path Names

Resources for learning Python

- **Books**
  - ***Learn to Program Using Python***
  - ***Learning Python***
  - ***The Quick Python Book***
  - ***Python, Essential Reference***



- **Web sites**
  - **The Python Foundation (www.python.org): Tutorials, documentation**
- **ESRI Instructor-led course**
  - **Introduction to Geoprocessing Scripts Using Python**
- ***Writing Geoprocessing Scripts with ArcGIS .pdf***
- **Online help**

Learning paths

- **Learning Guide**
  - **Learning paths organized by software and topic**
- **Learning options**
  - **Instructor-led courses**
  - **Virtual Campus courses**
  - **Training seminars**
  - **Web workshops**

Software support resources

- **ESRI Support Center**
  a gateway to resources

  `http://support.esri.com`

- **Knowledge Base**
  - **Technical articles**
  - **White papers**
  - **System requirements**
- **Downloads**
  - **Patches and service packs**
  - **Data models**
  - **ArcScripts**
- **User Forums**
  - **Discussion groups**
  - **E-mail lists**

Thank you for Attending!